Amazon GuardDuty User Guide

Amazon GuardDuty

Amazon GuardDuty: Amazon GuardDuty User Guide

Table of Contents

What is GuardDuty? 1
Features of GuardDuty 2
PCI DSS Compliance 5
Pricing in GuardDuty 5
Using GuardDuty 30-day free trial 6
Using Malware Protection for S3 with 12-month Free Tier 8
Accessing GuardDuty 8
Concepts and key terms 10
Getting started 15
Before you begin 15
Step 1: Enable Amazon GuardDuty 17
Step 2: Generate sample findings and explore basic operations 19
Step 3: Configure exporting GuardDuty findings to an Amazon S3 bucket 20
Step 4: Set up GuardDuty finding alerts through SNS 26
Next steps 28
Foundational data sources 30
AWS CloudTrail management events 30
How GuardDuty handles AWS CloudTrail global events 31
VPC Flow Logs 32
Route53 Resolver DNS query logs 33
Extended Threat Detection 34
Attack sequence threat scenario examples 34
How Extended Threat Detection works 35
Enabling protection plans to maximize threat detection 36
Detecting attack sequences in Amazon EKS clusters 36
Detecting attack sequences in Amazon S3 buckets 38
Extended Threat Detection in GuardDuty console 38
Understanding and managing attack sequence findings 38
Additional resources 39
EKS Protection 40
EKS audit logs in EKS Protection 41
Enabling EKS Protection in multiple-account environments 41
Enabling EKS Protection for a standalone account 48
S3 Protection 50

AWS CloudTrail data events for S3 ... 51
How GuardDuty uses CloudTrail data events for S3 ... 51
GuardDuty using CloudTrail data events for S3 for attack sequences ... 51
Enabling S3 Protection in multiple-account environments ... 52
Enabling S3 Protection for a standalone account ... 59
Runtime Monitoring ... 61
How it works ... 62
With Amazon EKS clusters ... 63
With Amazon EC2 instances ... 68
With Fargate (Amazon ECS only) ... 71
After you enable Runtime Monitoring ... 73
30-day free trial ... 74
I am using GuardDuty trial period or I have never enabled EKS Runtime Monitoring ... 75
I enabled EKS Runtime Monitoring prior to the launch of Runtime Monitoring ... 75
Prerequisites ... 76
For EC2 instance ... 77
For Fargate (ECS only) cluster ... 82
For EKS cluster ... 88
Enabling Runtime Monitoring ... 92
Enabling Runtime Monitoring for multiple-account environments ... 92
Enabling Runtime Monitoring for a standalone account ... 97
Managing GuardDuty security agents ... 98
Automated agent on Amazon EC2 resource ... 98
Manual agent management for Amazon EC2 resource ... 111
Automated agent on Fargate (Amazon ECS only) ... 126
Automated agent on Amazon EKS resource ... 161
Manual agent management for Amazon EKS cluster ... 199
Configure EKS add-on parameters ... 207
Validating VPC endpoint configuration ... 210
Runtime coverage issues and troubleshooting ... 212
Coverage and troubleshooting for Amazon EC2 resources ... 212
Coverage and troubleshooting for Amazon ECS clusters ... 227
Coverage and troubleshooting for Amazon EKS clusters ... 240
Setting up CPU and memory monitoring ... 256
Using shared VPC ... 257
How it works ... 258

- Prerequisites ... 259
- Using IaC with automated agents ... 260
  - IaC resource dependency graph overview ... 261
  - Common issue - Deleting resources in IaC ... 261
- Collected runtime event types ... 262
  - Process events ... 263
  - Container events ... 265
  - AWS Fargate (Amazon ECS only) task events ... 265
  - Kubernetes pod events ... 266
  - Domain Name System (DNS) events ... 266
  - Open events ... 267
  - Load module event ... 268
  - Mprotect events ... 268
  - Mount events ... 268
  - Link events ... 269
  - Symlink events ... 269
  - Dup events ... 269
  - Memory map event ... 270
  - Socket events ... 271
  - Connect events ... 271
  - Process VM Readv events ... 272
  - Process VM Writev events ... 272
  - Process trace (Ptrace) events ... 273
  - Bind events ... 273
  - Listen events ... 274
  - Rename events ... 275
  - Set user ID (UID) events ... 275
  - Chmod events ... 275
- Amazon ECR repository hosting GuardDuty agent ... 276
- Security agents on same host ... 288
  - Overview ... 288
  - Impact ... 288
  - How GuardDuty handles multiple agents ... 289
- EKS Runtime Monitoring ... 289
  - Configuring EKS Runtime Monitoring for multiple-account environments (API) ... 290
  - Configuring EKS Runtime Monitoring for a standalone account (API) ... 329

Migrating from EKS Runtime Monitoring to Runtime Monitoring ... 335
GuardDuty security agent release versions ... 339
Additional resources - next steps ... 368
Disabling, uninstalling, and resource cleanup ... 369
Uninstalling security agent manually for Amazon EC2 resources ... 371
Cleaning up security agent resources ... 373
Malware Protection for EC2 ... 375
Comparing GuardDuty-initiated malware scan and On-demand malware scan ... 376
How GuardDuty scans EBS volumes for malware detection ... 378
Supported EBS volumes ... 380
Modify default KMS key ID ... 380
Set up snapshot retention and EC2 scan coverage ... 381
Snapshots retention ... 382
Scan options with user-defined tags ... 383
Global `GuardDutyExcluded` tag ... 387
GuardDuty-initiated malware scan ... 387
30-day free trial ... 388
Enabling GuardDuty-initiated malware scan in multiple-account environments ... 389
Enabling GuardDuty-initiated malware scan for a standalone account ... 399
Findings that invoke GuardDuty-initiated malware scan ... 400
On-demand malware scan ... 403
How On-demand malware scan works ... 404
Starting On-demand malware scan ... 404
Re-scanning previously scanned Amazon EC2 instance ... 407
Monitoring malware scan statuses and results ... 408
GuardDuty service account ... 410
Quotas in Malware Protection for EC2 ... 413
Malware Protection for S3 ... 417
Pricing and usage cost ... 419
Reviewing usage cost ... 420
How it works ... 420
Overview ... 420
IAM role permissions ... 420
Optional tagging of objects based on scan result ... 421
Process after you enable Malware Protection for S3 for a bucket ... 421
Capabilities of Malware Protection for S3 ... 424

- (Optional) Get started with Malware Protection for S3 only (console) ... 425
- Configuring Malware Protection for S3 for your bucket ... 426
  - Enabling Malware Protection for S3 threat detection for your bucket ... 427
  - IAM role permissions ... 433
  - Troubleshooting IAM role permissions error ... 438
- Steps after enabling Malware Protection for S3 ... 439
- Using tag-based access control (TBAC) ... 440
  - Adding TBAC on S3 bucket resource ... 441
- View and understand protected bucket status ... 443
- Troubleshooting Malware Protection plan status ... 444
  - EventBridge notification is disabled for this S3 bucket ... 445
  - EventBridge managed rule to receive S3 bucket events is missing ... 446
  - S3 bucket no longer exists ... 446
  - Unable to put test object ... 447
- Monitoring S3 object scans ... 448
  - S3 object potential scan status and result status ... 448
  - Using Amazon EventBridge ... 450
  - Using S3 Object Tags ... 459
  - Using CloudWatch alarms and metrics ... 460
- Editing Malware Protection plan for a protected bucket ... 463
- Disabling Malware Protection for S3 for a protected bucket ... 465
- Supportability of Amazon S3 features ... 467
- Malware Protection for S3 quotas ... 473

RDS Protection ... 476
- Supported databases ... 477
- RDS login activity ... 478
- Enabling RDS Protection in multiple-account environments ... 479
- Enabling RDS Protection for a standalone account ... 485

Lambda Protection ... 487
- Lambda Network Activity Monitoring ... 488
- Enabling Lambda Protection in multiple-account environments ... 488
- Enabling Lambda Protection for a standalone account ... 495

Protecting AI workloads ... 497

Multiple accounts in GuardDuty ... 498
- Administrator account and member account relationships ... 498
- Managing accounts with AWS Organizations ... 503

Considerations and recommendations 504
Permissions required to designate a delegated GuardDuty administrator account 506
Designating delegated GuardDuty administrator account 507
Setting organization auto-enable preferences 509
Adding members to the organization 513
(Optional) Enable protection plans for existing member accounts 515
Continually managing your member accounts within GuardDuty 516
Suspending GuardDuty for member account 517
Disassociating (removing) member account from administrator account 519
Deleting member accounts from GuardDuty organization 520
Changing the delegated GuardDuty administrator account 522
Managing accounts by invitation 524
Adding accounts by invitation 525
Consolidating administrator accounts under a single organization 530
GuardDuty considerations for Export CSV option in accounts 533
Finding types 534
EC2 finding types 534
Backdoor:EC2/C&CActivity.B 536
Backdoor:EC2/C&CActivity.B!DNS 537
Backdoor:EC2/DenialOfService.Dns 538
Backdoor:EC2/DenialOfService.Tcp 539
Backdoor:EC2/DenialOfService.Udp 539
Backdoor:EC2/DenialOfService.UdpOnTcpPorts 540
Backdoor:EC2/DenialOfService.UnusualProtocol 541
Backdoor:EC2/Spambot 541
Behavior:EC2/NetworkPortUnusual 542
Behavior:EC2/TrafficVolumeUnusual 542
CryptoCurrency:EC2/BitcoinTool.B 543
CryptoCurrency:EC2/BitcoinTool.B!DNS 544
DefenseEvasion:EC2/UnusualDNSResolver 544
DefenseEvasion:EC2/UnusualDoHActivity 545
DefenseEvasion:EC2/UnusualDoTActivity 545
Impact:EC2/AbusedDomainRequest.Reputation 546
Impact:EC2/BitcoinDomainRequest.Reputation 546
Impact:EC2/MaliciousDomainRequest.Reputation 547
Impact:EC2/MaliciousDomainRequest.Custom 548

Impact:EC2/PortSweep ... 548
Impact:EC2/SuspiciousDomainRequest.Reputation ... 549
Impact:EC2/WinRMBruteForce ... 549
Recon:EC2/PortProbeEMRUnprotectedPort ... 550
Recon:EC2/PortProbeUnprotectedPort ... 551
Recon:EC2/Portscan ... 552
Trojan:EC2/BlackholeTraffic ... 552
Trojan:EC2/BlackholeTraffic!DNS ... 553
Trojan:EC2/DGADomainRequest.B ... 554
Trojan:EC2/DGADomainRequest.C!DNS ... 554
Trojan:EC2/DNSDataExfiltration ... 555
Trojan:EC2/DriveBySourceTraffic!DNS ... 556
Trojan:EC2/DropPoint ... 556
Trojan:EC2/DropPoint!DNS ... 557
Trojan:EC2/PhishingDomainRequest!DNS ... 557
UnauthorizedAccess:EC2/MaliciousIPCaller.Custom ... 558
UnauthorizedAccess:EC2/MetadataDNSRebind ... 558
UnauthorizedAccess:EC2/RDPBruteForce ... 559
UnauthorizedAccess:EC2/SSHBruteForce ... 560
UnauthorizedAccess:EC2/TorClient ... 561
UnauthorizedAccess:EC2/TorRelay ... 562
IAM finding types ... 562
CredentialAccess:IAMUser/AnomalousBehavior ... 563
DefenseEvasion:IAMUser/AnomalousBehavior ... 564
Discovery:IAMUser/AnomalousBehavior ... 565
Exfiltration:IAMUser/AnomalousBehavior ... 566
Impact:IAMUser/AnomalousBehavior ... 566
InitialAccess:IAMUser/AnomalousBehavior ... 567
PenTest:IAMUser/KaliLinux ... 568
PenTest:IAMUser/ParrotLinux ... 568
PenTest:IAMUser/PentooLinux ... 569
Persistence:IAMUser/AnomalousBehavior ... 569
Policy:IAMUser/RootCredentialUsage ... 570
Policy:IAMUser/ShortTermRootCredentialUsage ... 571
PrivilegeEscalation:IAMUser/AnomalousBehavior ... 571
Recon:IAMUser/MaliciousIPCaller ... 572

- Recon:IAMUser/MaliciousIPCaller.Custom ... 572
- Recon:IAMUser/TorIPCaller ... 573
- Stealth:IAMUser/CloudTrailLoggingDisabled ... 573
- Stealth:IAMUser/PasswordPolicyChange ... 574
- UnauthorizedAccess:IAMUser/ConsoleLoginSuccess.B ... 575
- UnauthorizedAccess:IAMUser/InstanceCredentialExfiltration.InsideAWS ... 575
- UnauthorizedAccess:IAMUser/InstanceCredentialExfiltration.OutsideAWS ... 577
- UnauthorizedAccess:IAMUser/MaliciousIPCaller ... 579
- UnauthorizedAccess:IAMUser/MaliciousIPCaller.Custom ... 579
- UnauthorizedAccess:IAMUser/TorIPCaller ... 579

Attack sequence finding types ... 580
- AttackSequence:EKS/CompromisedCluster ... 581
- AttackSequence:IAM/CompromisedCredentials ... 582
- AttackSequence:S3/CompromisedData ... 582

S3 Protection finding types ... 583
- Discovery:S3/AnomalousBehavior ... 584
- Discovery:S3/MaliciousIPCaller ... 585
- Discovery:S3/MaliciousIPCaller.Custom ... 586
- Discovery:S3/TorIPCaller ... 586
- Exfiltration:S3/AnomalousBehavior ... 587
- Exfiltration:S3/MaliciousIPCaller ... 588
- Impact:S3/AnomalousBehavior.Delete ... 588
- Impact:S3/AnomalousBehavior.Permission ... 589
- Impact:S3/AnomalousBehavior.Write ... 590
- Impact:S3/MaliciousIPCaller ... 591
- PenTest:S3/KaliLinux ... 591
- PenTest:S3/ParrotLinux ... 592
- PenTest:S3/PentooLinux ... 592
- Policy:S3/AccountBlockPublicAccessDisabled ... 593
- Policy:S3/BucketAnonymousAccessGranted ... 593
- Policy:S3/BucketBlockPublicAccessDisabled ... 594
- Policy:S3/BucketPublicAccessGranted ... 595
- Stealth:S3/ServerAccessLoggingDisabled ... 596
- UnauthorizedAccess:S3/MaliciousIPCaller.Custom ... 596
- UnauthorizedAccess:S3/TorIPCaller ... 597

EKS Protection finding types ... 597

CredentialAccess:Kubernetes/MaliciousIPCaller 599
CredentialAccess:Kubernetes/MaliciousIPCaller.Custom 600
CredentialAccess:Kubernetes/SuccessfulAnonymousAccess 600
CredentialAccess:Kubernetes/TorIPCaller 601
DefenseEvasion:Kubernetes/MaliciousIPCaller 602
DefenseEvasion:Kubernetes/MaliciousIPCaller.Custom 602
DefenseEvasion:Kubernetes/SuccessfulAnonymousAccess 603
DefenseEvasion:Kubernetes/TorIPCaller 604
Discovery:Kubernetes/MaliciousIPCaller 604
Discovery:Kubernetes/MaliciousIPCaller.Custom 605
Discovery:Kubernetes/SuccessfulAnonymousAccess 606
Discovery:Kubernetes/TorIPCaller 607
Execution:Kubernetes/ExecInKubeSystemPod 607
Impact:Kubernetes/MaliciousIPCaller 608
Impact:Kubernetes/MaliciousIPCaller.Custom 609
Impact:Kubernetes/SuccessfulAnonymousAccess 609
Impact:Kubernetes/TorIPCaller 610
Persistence:Kubernetes/ContainerWithSensitiveMount 611
Persistence:Kubernetes/MaliciousIPCaller 611
Persistence:Kubernetes/MaliciousIPCaller.Custom 612
Persistence:Kubernetes/SuccessfulAnonymousAccess 613
Persistence:Kubernetes/TorIPCaller 613
Policy:Kubernetes/AdminAccessToDefaultServiceAccount 614
Policy:Kubernetes/AnonymousAccessGranted 615
Policy:Kubernetes/ExposedDashboard 615
Policy:Kubernetes/KubeflowDashboardExposed 616
PrivilegeEscalation:Kubernetes/PrivilegedContainer 616
CredentialAccess:Kubernetes/AnomalousBehavior.SecretsAccessed 617
PrivilegeEscalation:Kubernetes/AnomalousBehavior.RoleBindingCreated 618
Execution:Kubernetes/AnomalousBehavior.ExecInPod 619
PrivilegeEscalation:Kubernetes/AnomalousBehavior.WorkloadDeployed!
PrivilegedContainer 620
Persistence:Kubernetes/AnomalousBehavior.WorkloadDeployed!
ContainerWithSensitiveMount 621
Execution:Kubernetes/AnomalousBehavior.WorkloadDeployed 622
PrivilegeEscalation:Kubernetes/AnomalousBehavior.RoleCreated 623

Discovery:Kubernetes/AnomalousBehavior.PermissionChecked ... 624
Runtime Monitoring finding types ... 625
CryptoCurrency:Runtime/BitcoinTool.B ... 626
Backdoor:Runtime/C&CActivity.B ... 627
UnauthorizedAccess:Runtime/TorRelay ... 628
UnauthorizedAccess:Runtime/TorClient ... 629
Trojan:Runtime/BlackholeTraffic ... 630
Trojan:Runtime/DropPoint ... 630
CryptoCurrency:Runtime/BitcoinTool.B!DNS ... 631
Backdoor:Runtime/C&CActivity.B!DNS ... 632
Trojan:Runtime/BlackholeTraffic!DNS ... 633
Trojan:Runtime/DropPoint!DNS ... 634
Trojan:Runtime/DGADomainRequest.C!DNS ... 634
Trojan:Runtime/DriveBySourceTraffic!DNS ... 635
Trojan:Runtime/PhishingDomainRequest!DNS ... 636
Impact:Runtime/AbusedDomainRequest.Reputation ... 637
Impact:Runtime/BitcoinDomainRequest.Reputation ... 638
Impact:Runtime/MaliciousDomainRequest.Reputation ... 639
Impact:Runtime/SuspiciousDomainRequest.Reputation ... 639
UnauthorizedAccess:Runtime/MetadataDNSRebind ... 640
Execution:Runtime/NewBinaryExecuted ... 641
PrivilegeEscalation:Runtime/DockerSocketAccessed ... 642
PrivilegeEscalation:Runtime/RuncContainerEscape ... 643
PrivilegeEscalation:Runtime/CGroupsReleaseAgentModified ... 644
DefenseEvasion:Runtime/ProcessInjection.Proc ... 645
DefenseEvasion:Runtime/ProcessInjection.Ptrace ... 645
DefenseEvasion:Runtime/ProcessInjection.VirtualMemoryWrite ... 646
Execution:Runtime/ReverseShell ... 647
DefenseEvasion:Runtime/FilelessExecution ... 647
Impact:Runtime/CryptoMinerExecuted ... 648
Execution:Runtime/NewLibraryLoaded ... 649
PrivilegeEscalation:Runtime/ContainerMountsHostDirectory ... 649
PrivilegeEscalation:Runtime/UserfaultfdUsage ... 650
Execution:Runtime/SuspiciousTool ... 651
Execution:Runtime/SuspiciousCommand ... 651
DefenseEvasion:Runtime/SuspiciousCommand ... 652

- DefenseEvasion:Runtime/PtraceAntiDebugging ... 653
- Execution:Runtime/MaliciousFileExecuted ... 654
- Execution:Runtime/SuspiciousShellCreated ... 654
- PrivilegeEscalation:Runtime/ElevationToRoot ... 655
- Discovery:Runtime/SuspiciousCommand ... 656
- Persistence:Runtime/SuspiciousCommand ... 657
- PrivilegeEscalation:Runtime/SuspiciousCommand ... 658

Malware Protection for EC2 finding types ... 658
- Execution:EC2/MaliciousFile ... 659
- Execution:ECS/MaliciousFile ... 660
- Execution:Kubernetes/MaliciousFile ... 660
- Execution:Container/MaliciousFile ... 661
- Execution:EC2/SuspiciousFile ... 661
- Execution:ECS/SuspiciousFile ... 662
- Execution:Kubernetes/SuspiciousFile ... 662
- Execution:Container/SuspiciousFile ... 663

Malware Protection for S3 finding type ... 664
- Object:S3/MaliciousFile ... 664

RDS Protection finding types ... 665
- CredentialAccess:RDS/AnomalousBehavior.SuccessfulLogin ... 665
- CredentialAccess:RDS/AnomalousBehavior.FailedLogin ... 666
- CredentialAccess:RDS/AnomalousBehavior.SuccessfulBruteForce ... 667
- CredentialAccess:RDS/MaliciousIPCaller.SuccessfulLogin ... 668
- CredentialAccess:RDS/MaliciousIPCaller.FailedLogin ... 669
- Discovery:RDS/MaliciousIPCaller ... 669
- CredentialAccess:RDS/TorIPCaller.SuccessfulLogin ... 670
- CredentialAccess:RDS/TorIPCaller.FailedLogin ... 671
- Discovery:RDS/TorIPCaller ... 671

Lambda Protection finding types ... 672
- Backdoor:Lambda/C&CActivity.B ... 672
- CryptoCurrency:Lambda/BitcoinTool.B ... 673
- Trojan:Lambda/BlackholeTraffic ... 674
- Trojan:Lambda/DropPoint ... 674
- UnauthorizedAccess:Lambda/MaliciousIPCaller.Custom ... 675
- UnauthorizedAccess:Lambda/TorClient ... 675
- UnauthorizedAccess:Lambda/TorRelay ... 676

- Retired finding types ... 676
  - Exfiltration:S3/ObjectRead.Unusual ... 677
  - Impact:S3/PermissionsModification.Unusual ... 678
  - Impact:S3/ObjectDelete.Unusual ... 679
  - Discovery:S3/BucketEnumeration.Unusual ... 679
  - Persistence:IAMUser/NetworkPermissions ... 680
  - Persistence:IAMUser/ResourcePermissions ... 681
  - Persistence:IAMUser/UserPermissions ... 681
  - PrivilegeEscalation:IAMUser/AdministrativePermissions ... 682
  - Recon:IAMUser/NetworkPermissions ... 683
  - Recon:IAMUser/ResourcePermissions ... 684
  - Recon:IAMUser/UserPermissions ... 684
  - ResourceConsumption:IAMUser/ComputeResources ... 685
  - Stealth:IAMUser/LoggingConfigurationModified ... 686
  - UnauthorizedAccess:IAMUser/ConsoleLogin ... 686
  - UnauthorizedAccess:EC2/TorIPCaller ... 687
  - Backdoor:EC2/XORDDOS ... 688
  - Behavior:IAMUser/InstanceLaunchUnusual ... 688
  - CryptoCurrency:EC2/BitcoinTool.A ... 688
  - UnauthorizedAccess:IAMUser/UnusualASNCaller ... 689
- GuardDuty finding types by potentially impacted resources ... 689
- GuardDuty active finding types ... 690

Understanding and generating findings ... 711

- GuardDuty finding format ... 712
  - Threat Purposes ... 713
- GuardDuty malware detection scan engine ... 716
- Sample findings ... 717
  - Generating sample findings through the GuardDuty console or API ... 717
- Test GuardDuty findings ... 719
  - Considerations ... 719
  - GuardDuty findings tester script can generate ... 721
  - Step 1 - Prerequisites ... 723
  - Step 2 - Deploy AWS resources ... 724
  - Step 3 - Run tester scripts ... 725
  - Step 4 - Clean up AWS test resources ... 728
  - Troubleshooting common issues ... 728

Findings page in GuardDuty console 730
Navigating Findings page 731
Findings severity levels 732
Critical severity 733
High severity 733
Medium severity 733
Low severity 734
Finding details 734
Finding overview 735
Resource 736
Attack sequence finding details 743
RDS database (DB) user details 750
Runtime Monitoring finding details 750
EBS volumes scan details 752
Malware Protection for EC2 finding details 753
Malware Protection for S3 finding details 754
Action 755
Actor or Target 757
Geolocation details 758
Additional information 758
Evidence 758
Anomalous behavior 759
GuardDuty finding aggregation 764
Managing GuardDuty findings 765
GuardDuty Summary dashboard 766
Overview 767
Findings 768
Most common finding types 769
Findings by severity 769
Accounts with most findings 769
Resources with findings 769
Least occurring findings 770
Protection plans coverage 770
Filtering GuardDuty findings 771
Creating and saving filter set in the GuardDuty console 772
Creating and saving filter set by using GuardDuty API and CLI 774

- Property filters in GuardDuty ... 776
- Suppression rules ... 783
  - ... 783
  - Using suppression rules with Extended Threat Detection ... 784
  - Common use cases for suppression rules and examples ... 784
  - Creating suppression rules ... 788
  - Deleting suppression rules ... 791
  - ... 789
- Entity lists and IP address lists ... 792
  - Understanding entity lists and IP address lists ... 792
  - Important considerations for GuardDuty lists ... 793
  - List formats ... 794
  - Understanding list statuses ... 800
  - Setting up prerequisites for entity lists and IP address lists ... 801
  - Adding and activating an entity list or IP list ... 803
  - Updating an entity list or IP address list ... 808
  - De-activating entity list or IP address list ... 813
  - Deleting entity list or IP address list ... 816
- Exporting generated findings to Amazon S3 ... 819
  - Considerations ... 819
  - Step 1 – Permissions required to export findings ... 820
  - Step 2 – Attaching policy to your KMS key ... 821
  - Step 3 – Attaching policy to Amazon S3 bucket ... 823
  - Step 4 - Exporting findings to an S3 bucket (Console) ... 827
  - Step 5 – Frequency for exporting findings ... 828
- Processing findings with EventBridge ... 828
  - EventBridge notification frequency in GuardDuty ... 829
  - Set up an Amazon SNS topic and endpoint ... 830
  - Using EventBridge with GuardDuty ... 831
  - Creating an EventBridge rule ... 833
  - EventBridge rule for multi-account environments ... 840
- Understanding CloudWatch Logs and reasons for skipping resources ... 841
  - Auditing CloudWatch Logs in GuardDuty Malware Protection for EC2 ... 841
  - GuardDuty Malware Protection for EC2 log retention ... 843
  - Reasons for skipping resource ... 843
- Reporting false positive EC2 malware scan result ... 847

Reporting false positive S3 object scan result 848
Remediating findings 850
Remediating a potentially compromised Amazon EC2 instance 850
Remediating a potentially compromised S3 bucket 852
Recommendations based on specific S3 bucket access needs 853
Remediating a potentially malicious S3 object 854
Remediating a potentially compromised ECS cluster 855
Remediating potentially compromised AWS credentials 855
Remediating a potentially compromised standalone container 857
Remediating EKS Protection findings 858
Potential configuration issues 859
Remediating potentially compromised Kubernetes users 859
Remediating potentially compromised Kubernetes pods 862
Remediating potentially compromised container images 864
Remediating potentially compromised Kubernetes nodes 864
Remediating Runtime Monitoring findings 865
Remediating compromised container images 867
Remediating a potentially compromised database 867
Remediating potentially compromised database with successful login events 868
Remediating potentially compromised database with failed login events 869
Remediating potentially compromised credentials 870
Restrict network access 870
Remediating a potentially compromised Lambda function 871
Estimating usage cost 872
Understanding how GuardDuty calculates usage costs 872
..... 873
Runtime Monitoring – How VPC flow logs from EC2 instances impact usage cost 873
How GuardDuty estimates usage cost for CloudTrail events 874
Reviewing estimated usage cost 874
Feature names for protection plans in API 877
Change from data sources to features 877
GuardDuty API changes 877
Features compared to data sources 878
Understanding how APIs with features work 878
Incorporating feature changes in APIs 879
Mapped GuardDuty feature 880

Security ... 882
Data protection ... 883
Encryption at rest ... 883
Encryption in transit ... 884
Opting out of using your data for service improvement ... 884
Logging with CloudTrail ... 886
GuardDuty information in CloudTrail ... 886
GuardDuty control plane events in CloudTrail ... 887
GuardDuty data events in CloudTrail ... 887
Example: GuardDuty log file entries ... 888
Identity and Access Management ... 891
Audience ... 891
Authenticating with identities ... 892
Managing access using policies ... 895
How Amazon GuardDuty works with IAM ... 898
Identity-based policy examples ... 904
Using service-linked roles ... 913
AWS managed policies ... 941
Troubleshooting ... 950
Compliance validation ... 952
Resilience ... 953
Infrastructure security ... 953
VPC endpoints (AWS PrivateLink) ... 954
Considerations for GuardDuty VPC endpoints ... 954
Creating an interface VPC endpoint for GuardDuty ... 954
Creating a VPC endpoint policy for GuardDuty ... 955
Shared subnets ... 955
Integration with AWS security services ... 956
Integrating GuardDuty with AWS Security Hub ... 956
Integrating GuardDuty with Amazon Detective ... 956
AWS Security Hub integration ... 956
How Amazon GuardDuty sends findings to AWS Security Hub ... 957
Viewing GuardDuty findings in AWS Security Hub ... 958
Enabling and configuring the integration ... 977
Using GuardDuty controls in Security Hub ... 977
Stopping the publication of findings to Security Hub ... 977

- Amazon Detective integration ... 977
  - Enabling the integration ... 978
  - Pivoting to Amazon Detective from a GuardDuty finding ... 978
  - Using the integration with a GuardDuty multi-account environment ... 979

Suspending or disabling ... 980

GuardDuty announcements ... 982

- Amazon SNS message format ... 988

GuardDuty quotas ... 993

Troubleshooting ... 999

- Exporting findings to Amazon S3 - access error ... 999
- Malware Protection for EC2 issues ... 1000
  - Missing required AWS Organizations management permission when enabling GuardDuty-initiated malware scan ... 1000
  - I am initiating an On-demand malware scan but it results in a missing required permissions error. ... 1000
  - I receive an `iam:GetRole` error while working with Malware Protection for EC2. ... 1000
  - I am a GuardDuty administrator account who needs to enable GuardDuty-initiated malware scan but doesn't use AWS managed policy: AmazonGuardDutyFullAccess to manage GuardDuty. ... 1001
- Runtime Monitoring issues ... 1001
  - Runtime coverage issues ... 1001
  - Troubleshooting out of memory error ... 1001
  - My AWS Step Functions workflow is failing unexpectedly ... 1002
- Other troubleshooting issues ... 1002

Regions and endpoints ... 1004

- Region-specific feature availability ... 1004

Legacy actions and parameters ... 1006

Document history ... 1008

- Earlier updates ... 1086

What is Amazon GuardDuty?

Amazon GuardDuty is a threat detection service that continuously monitors, analyzes, and processes AWS data sources and logs in your AWS environment. GuardDuty uses threat intelligence feeds, such as lists of malicious IP addresses and domains, file hashes, and machine learning (ML) models to identify suspicious and potentially malicious activity in your AWS environment. The following list provides an overview of potential threat scenarios that GuardDuty can help you detect:

- Compromised and exfiltrated AWS credentials.
- Data exfiltration and destruction that can lead to a ransomware event. Unusual patterns of login events in the supported engine versions of Amazon Aurora and Amazon RDS databases, that indicate anomalous behavior.
- Unauthorized cryptomining activity in your Amazon Elastic Compute Cloud (Amazon EC2) instances and container workloads.
- Presence of malware in your Amazon EC2 instances and container workloads, and newly uploaded files in your Amazon Simple Storage Service (Amazon S3) buckets.
- Operating system-level, networking, and file events that indicate unauthorized behavior on your Amazon Elastic Kubernetes Service (Amazon EKS) clusters, Amazon Elastic Container Service (Amazon ECS) - AWS Fargate tasks, and Amazon EC2 instances and container workloads.

The following video provides an overview of how GuardDuty helps you detect threats in your AWS environment.

What is Amazon GuardDuty

Contents

- Features of GuardDuty
- PCI DSS Compliance
- Pricing in GuardDuty
- Accessing GuardDuty

Features of GuardDuty

Here are some of the key ways in which Amazon GuardDuty can help you monitor, detect, and manage potential threats in your AWS environment.

Continuously monitors specific data sources and event logs

- **Foundational threat detection –** When you enable GuardDuty in an AWS account, GuardDuty automatically starts ingesting the foundational data sources associated with that account. These data sources include AWS CloudTrail management events, VPC flow logs (from Amazon EC2 instances), and DNS logs. You don't need to enable anything else for GuardDuty to start analyzing and processing these data sources to generate associated security findings. For more information, see GuardDuty foundational data sources.
- **Extended Threat Detection –** This capability detects multi-stage attacks that span foundational data sources, multiple types of AWS resources, and time, within an AWS account. There might be multiple events in your account that, individually, don't present themselves as a clear threat. However, when these events are observed in a sequence that is indicative of a suspicious activity, GuardDuty identifies it as an attack sequence. GuardDuty notifies you by generating the associated attack sequence finding type to provide details about the observed attack sequence.

  With no additional cost associated with it, Extended Threat Detection is automatically enabled for each AWS account when they enable GuardDuty. This capability doesn't require you to enable any use-case focused protection plan. However, to increase the breadth of security to your Amazon S3 resources, GuardDuty recommends enabling S3 Protection in your account. This will help Extended Threat Detection to identify multi-stage attacks that potentially impact your Amazon S3 resources.

  For more information about how this capability works and what threat scenarios it covers, see GuardDuty Extended Threat Detection.
- **Use-case focused GuardDuty protection plans –** For enhanced threat detection visibility into the security of your AWS environment, GuardDuty offers dedicated protection plans that you can choose to enable. Protection plans help you monitor logs and events from other AWS services. These sources include EKS audit logs, RDS login activity, Amazon S3 data events in CloudTrail, EBS volumes, Runtime Monitoring across Amazon EKS, Amazon EC2, and Amazon ECS-Fargate, and Lambda network activity logs. GuardDuty consolidates these log and event sources under the term - Features. You can enable one or more dedicated protection plans in a supported AWS Region at any time. GuardDuty will start monitoring,

processing, and analyzing the activities based on which protection plan you enable. For more information about each protection plan and how it works, see the corresponding protection plan document.

Protection plan	Description
S3 Protection	Identifies potential security risks such as data exfiltration and destruction attempts in your Amazon S3 buckets.
EKS Protection	EKS Audit Log Monitoring analyzes Kubernetes audit logs from your Amazon EKS clusters for potentially suspicious and malicious activities.
Runtime Monitoring	Monitors and analyzes operating system-level events on your Amazon EKS, Amazon EC2, and Amazon ECS (includin g AWS Fargate), to detect potential runtime threats.
Malware Protection for EC2	Detects potential presence of malware by scanning the Amazon EBS volumes associated with your Amazon EC2 instances. There is an option to use this feature on-demand .
Malware Protection for S3	Detects potential presence of malware in the newly uploaded objects within your Amazon S3 buckets.
RDS Protection	Analyzes and profiles your RDS login activity for potential access threats to the supported Amazon Aurora and Amazon RDS databases.
Lambda Protection	Monitors Lambda network activity logs, starting with VPC flow logs, to detect threats to your AWS Lambda functions . Examples of these potential threats include cryptomining and communicating with malicious servers.

Enable Malware Protection for S3 independently

GuardDuty offers flexibility to use Malware Protection for S3 independently, without enabling the Amazon GuardDuty service. For more information about getting started with only Malware Protection for S3, see GuardDuty Malware Protection for S3. To use all other protection plans, you must enable the GuardDuty service.

Manage multiple-account environment

You can manage a multiple-account AWS environment by using either AWS Organizations (recommended) or legacy invitation method. For more information, see Multiple accounts in GuardDuty.

Generates security findings for detected threats

When GuardDuty detects potential security threats associated with your AWS resources, it starts generating security findings that provide information about the potentially compromised resource. After you enable GuardDuty in your account, generate Sample findings to view the associated Finding details. For a complete list of security findings, see GuardDuty finding types.

With GuardDuty, you can also use a tester script that generates specific GuardDuty security findings to understand how to review and respond to GuardDuty findings. For more information, see Test GuardDuty findings in dedicated accounts.

Assessing and managing security findings

GuardDuty consolidates your security findings across accounts and displays results in the Summary dashboard on the GuardDuty console. You can also retrieve findings through the AWS Security Hub API, AWS Command Line Interface, or AWS SDK. With a holistic view of your current security status, you can identify trends and potential issues, and take necessary remediation steps. For more information, see Managing GuardDuty findings.

Integrate with related AWS security services

To further help you analyze and investigate the security trends in your AWS environment, consider using the following AWS security-related services in combination with GuardDuty.

- **AWS Security Hub –** This service gives you a comprehensive view of the security state of your AWS resources and helps you check your AWS environment against security industry standards and best practices. It does this partly by consuming, aggregating, organizing, and prioritizing your security findings from multiple AWS services (including Amazon Macie) and supported AWS Partner Network (APN) products. Security Hub helps you analyze your security trends and identify the highest priority security issues across your AWS environment.

For information about using GuardDuty and Security Hub together, see Integrating GuardDuty with AWS Security Hub. To learn more about Security Hub, see the AWS Security Hub User Guide.

- **Amazon Detective –** This service helps you analyze, investigate, and quickly identify the root cause of security findings or suspicious activities. Detective automatically collects log data from your AWS resources. It then uses machine learning, statistical analysis, and graph theory to generate visualizations that help you to conduct faster and more efficient security investigations. The Detective prebuilt data aggregations, summaries, and context help you analyze and determine the nature and extent of potential security issues.

  For information about using GuardDuty and Detective together, see Integrating GuardDuty with Amazon Detective. To learn more about Detective, see the Amazon Detective User Guide.
- **Amazon EventBridge –** This service helps you receive notifications and respond to GuardDuty security findings in near-real time. GuardDuty creates an event when there is a change in the findings. You can choose how frequently you want to receive the notifications from EventBridge. For more information, see What is Amazon EventBridge in the *Amazon EventBridge User Guide*.

PCI DSS Compliance

GuardDuty supports the processing, storage, and transmission of credit card data by a merchant or service provider, and has been validated as being compliant with Payment Card Industry (PCI) Data Security Standard (DSS). For more information about PCI DSS, including how to request a copy of the AWS PCI Compliance Package, see PCI DSS Level 1.

For more information, see New third-party test compares Amazon GuardDuty to network intrusion detection systems in the *AWS Security Blog*.

Pricing in GuardDuty

This section focuses on the AWS Free Tier model that GuardDuty uses for various protection plans, and how you can view estimated and actual usage costs. If you are looking for the pricing details associated with all the protection plans across supported Regions, see GuardDuty pricing.

AWS Free Tier

AWS Free Tier helps you explore and try out AWS services free of charge up to specified limits for each service. There are three categories – 12 months free, always free, and short-term free trials. Amazon GuardDuty belongs to the short-term free trial category and offers a 30-day free trial. When you continue using GuardDuty after this free trial ends, you start incurring cost based on how you use this service.

[1]Exception to GuardDuty 30-day free trial

On-demand malware scan (under Malware Protection for EC2) and Malware Protection for S3 don't fall into the GuardDuty 30-day short term free trial category. Malware Protection for S3 falls into the 12 months free category of the AWS Free Tier whereas the On-demand malware scan follows a pay-as-you-use cost model. There is no 30-day free trial or a 12-month Free Tier cost model with On-demand malware scan.

Using GuardDuty 30-day free trial

When using GuardDuty for the first time in an AWS Region, your AWS account is automatically enrolled in a 30-day free trial in that Region. Some of the protection plans will also get enabled automatically and are included in the 30-day free trial. Because GuardDuty is a regional service, when you enable it for the first time in a different Region, your account will get a 30-day free trial of GuardDuty in that Region. When working with multiple accounts in a GuardDuty organization, each account gets its own 30-day free trial.

Use the following table to review which protection plans are enabled by default with GuardDuty, and their free trial availability.

Protection plan	Enabled by default with GuardDuty	Separate free trial availability[2]
EKS Protection	Yes	Yes
S3 Protection	Yes	Yes
Runtime Monitoring	No	Yes
Malware Protection for EC2 – GuardDuty	Yes	Yes

Protection plan	Enabled by default with GuardDuty	Separate free trial availability[2]
-initiated malware scan		
Malware Protection for EC2 – On-demand malware scan in GuardDuty	No	No[1]
GuardDuty Malware Protection for S3	No	No[1]
RDS Protection	Yes	Yes
Lambda Protection	Yes	Yes

[2]When you enable GuardDuty for the first time, protection plans (except Runtime Monitoring) are automatically enabled and included in the initial 30-day free trial. When an existing GuardDuty account enables a new protection plan after their initial GuardDuty free trial has expired, then that protection plan comes with its own 30-day free trial. For more information about free trials for protection plans, see the document associated with each protection plan.

View estimated usage cost during free trial – During 30-day free trial of GuardDuty and potentially a protection plan, GuardDuty provides estimated usage cost for your account. If you're a delegated GuardDuty administrator account, you can view the total estimated usage cost and account-level breakdown for all the member accounts that have enabled GuardDuty. For more information, see Estimating GuardDuty usage cost.

Usage cost after free trial ends – When you continue using GuardDuty or any of its protection plans after the free trial ends, you will start incurring associated usage costs. To view your bill, navigate to **Cost Explorer** in the the https://console.aws.amazon.com/costmanagement/ console. For more information about AWS account billing, see the AWS Billing User Guide.

Using Malware Protection for S3 with 12-month Free Tier

Malware Protection for S3 uses a Free Tier plan associated with your AWS accounts that are either new, have an ongoing free tier, or have an expired 12-month free tier. For more information, see Pricing and usage cost for Malware Protection for S3.

Accessing GuardDuty

Amazon GuardDuty is available in most AWS Regions. For a list of Regions where GuardDuty is currently available, see Regions and endpoints.

You can use GuardDuty in any of the following ways:

GuardDuty console

https://console.aws.amazon.com/guardduty/

The console is a browser-based interface to access and use GuardDuty. The GuardDuty console provides access to your GuardDuty account, data, and resources.

AWS Command Line Interface

With AWS Command Line Interface (AWS CLI), you can issue commands at your system's command line to perform GuardDuty tasks and AWS tasks. The AWS CLI commands are useful if you want to build scripts that perform tasks.

For information about installing and using AWS CLI, see AWS Command Line Interface User Guide. To view the available AWS CLI commands for GuardDuty, see AWS CLI Command Reference.

GuardDuty HTTPS API

You can access GuardDuty and AWS programmatically by using the GuardDuty HTTPS API, which lets you issue HTTPS requests directly to the service. For more information, see the Amazon GuardDuty API Reference.

AWS SDKs

AWS provides software development kits (SDKs) that consist of libraries and sample code for various programming languages and platforms (Java, Python, Ruby, .NET, iOS, Android, and more). The SDKs provide a convenient way to create programmatic access to GuardDuty. For information about the AWS SDKs, including how to download and install them, see Tools for Amazon Web Services.

Concepts and key terms in Amazon GuardDuty

As you get started with Amazon GuardDuty, you can benefit from learning about its concepts and associated key terms.

Account

A standard Amazon Web Services (AWS) account that contains your AWS resources. You can sign in to AWS with your account and enable GuardDuty.

You can also invite other accounts to enable GuardDuty and become associated with your AWS account in GuardDuty. If your invitations are accepted, your account is designated as the **administrator account** GuardDuty account, and the added accounts become your **member** accounts. You can then view and manage those accounts' GuardDuty findings on their behalf.

Users of the administrator account can configure GuardDuty as well as view and manage GuardDuty findings for their own account and all of their member accounts. For information about the number of member accounts that your administrator account can manage, see GuardDuty quotas.

Users of member accounts can configure GuardDuty as well as view and manage GuardDuty findings in their account (either through the GuardDuty management console or GuardDuty API). Users of member accounts can't view or manage findings in other members' accounts.

An AWS account can't be a GuardDuty administrator account and member account at the same time. An AWS account can accept only one membership invitation. Accepting a membership invitation is optional.

For more information, see Multiple accounts in Amazon GuardDuty.

Attack sequence

An attack sequence is a correlation of multiple events that, as observed by GuardDuty, happened in a specific sequence that matches the pattern of a suspicious activity. GuardDuty uses its Extended Threat Detection capability to detect these multi-stage attacks that span foundational data sources, AWS resources, and timeline, in your account.

The following list briefly explains the key terms associated with attack sequences:

- **Indicators –** Provides information as to why a sequence of events aligns with a potential suspicious activity.

- **Signals –** A signal is an API activity that GuardDuty observed, or an already detected GuardDuty finding in your account. By correlating the events that were observed in a specific sequence in your account, GuardDuty identifies an attack sequence.

  There are events in your account that are not indicative of a potential threat. GuardDuty considers them as **weak** signals. However, when weak signals and GuardDuty findings are observed in a specific sequence that, when correlated, align to a potentially suspicious activity, GuardDuty generates an attack sequence finding.
- **Endpoints –** Information about network endpoints that a threat actor potentially used in an attack sequence.

Detector

Amazon GuardDuty is a regional service. When you enable GuardDuty in a specific AWS Region, your AWS account gets associated with a detector ID. This 32-character alphanumeric ID is unique to your account in that Region. For example, when you enable GuardDuty for the same account in a different Region, your account will get associated with a different detector ID. The format of a detectorId is `12abc34d567e8fa901bc2d34e56789f0`.

All GuardDuty findings, accounts, and actions about managing findings and the GuardDuty service use detector ID to run an API operation.

To find the `detectorId` for your account and current Region, see the **Settings** page in the https://console.aws.amazon.com/guardduty/ console, or run the ListDetectors API.

> **Note**
>
> In multiple-account environments, all findings for member accounts roll up to the administrator account's detector.

Some GuardDuty functionality is configured through the detector, such as configuring CloudWatch Events notification frequency, and the enabling or disabling of optional protection plans for GuardDuty to process.

Using Malware Protection for S3 within GuardDuty

When you enable Malware Protection for S3 in an account where GuardDuty is enabled, the Malware Protection for S3 actions such as enabling, editing, and disabling a protected resource are not associated with the detector ID.

When you don't enable GuardDuty and choose the threat detection option Malware Protection for S3, there is no detector ID that gets created for your account.

Foundational data sources

The origin or location of a set of data. To detect an unauthorized or unexpected activity in your AWS environment. GuardDuty analyzes and processes data from AWS CloudTrail event logs, AWS CloudTrail management events, AWS CloudTrail data events for S3, VPC flow logs, DNS logs, see GuardDuty foundational data sources.

Feature

A feature object configured for your GuardDuty protection plan helps to detect an unauthorized or unexpected activity in your AWS environment. Each GuardDuty protection plan configures the corresponding feature object to analyze and process data. Some of the feature objects include EKS audit logs, RDS login activity monitoring, Lambda network activity logs, and EBS volumes. For more information, see Feature names for protection plans in GuardDuty API.

Finding

A potential security issue discovered by GuardDuty. For more information, see Understanding and generating Amazon GuardDuty findings.

Findings are displayed in the GuardDuty console and contain a detailed description of the security issue. You can also retrieve your generated findings by calling the GetFindings and ListFindings API operations.

You can also see your GuardDuty findings through Amazon CloudWatch events. GuardDuty sends findings to Amazon CloudWatch through HTTPS protocol. For more information, see Processing GuardDuty findings with Amazon EventBridge.

IAM role

This is the IAM role with the required permissions to scan the S3 object. When tagging scanned objects is enabled, the IAM PassRole permissions help GuardDuty add tags to the scanned object.

Malware Protection plan resource

After you enable Malware Protection for S3 for a bucket, GuardDuty creates a Malware Protection for EC2 plan resource. This resource is associated with Malware Protection for EC2 plan ID, a unique identifier for your protected bucket. Use Malware Protection plan resource to perform API operations on a protected resource.

Protected bucket (protected resource)

An Amazon S3 bucket is considered to be protected when you enable Malware Protection for S3 for this bucket and its protection status changes to **Active**.

GuardDuty supports only an S3 bucket as a protected resource.

Protection status

The status associated with your Malware Protection plan resource. After you enable Malware Protection for S3 for your bucket, this status represents whether or not your bucket is set up correctly.

S3 object prefix

In an Amazon Simple Storage Service (Amazon S3) bucket, you can use prefixes to organize your storage. A prefix is a logical grouping of the objects in an S3 bucket. For more information, see Organizing and listing objects in the *Amazon S3 User Guide*.

Scan options

When GuardDuty Malware Protection for EC2 is enabled, it allows you to specify which Amazon EC2 instances and Amazon Elastic Block Store(EBS) volumes to scan or skip. This feature lets you add the existing tags that are associated with your EC2 instances and EBS volume to either an inclusion tags list or exclusion tags list. The resources associated to the tags that you add to an inclusion tags list, are scanned for malware, and those added to an exclusion tags list are not scanned. For more information, see Scan options with user-defined tags.

Snapshots retention

When GuardDuty Malware Protection for EC2 is enabled, it provides an option to retain the snapshots of your EBS volumes in your AWS account. GuardDuty generates the replica EBS volumes based on the snapshots of your EBS volumes. You can retain the snapshots of your EBS volumes only if the Malware Protection for EC2 scan detects malware in the replica EBS volumes. If no malware is detected in the replica EBS volumes, GuardDuty automatically deletes the snapshots of your EBS volumes, irrespective of the snapshots retention setting. For more information, see Snapshots retention.

Suppression rule

Suppression rules allow you to create very specific combinations of attributes to suppress findings. For example, you can define a rule through the GuardDuty filter to auto-archive `Recon:EC2/Portscan` from only those instances in a specific VPC, running a specific AMI, or with a specific EC2 tag. This rule would result in port scan findings being automatically archived from the instances that meet the criteria. However, it still allows alerting if GuardDuty detects those instances conducting other malicious activity, such as crypto-currency mining.

Suppression rules defined in the GuardDuty administrator account apply to the GuardDuty member accounts. GuardDuty member accounts can't modify suppression rules.

With suppression rules, GuardDuty still generates all findings. Suppression rules provide suppression of findings while maintaining a complete and immutable history of all activity.

Typically suppression rules are used to hide findings that you have determined as false positives for your environment, and reduce the noise from low-value findings so you can focus on larger threats. For more information, see Suppression rules in GuardDuty.

Trusted IP list

A list of trusted IP addresses for highly secure communication with your AWS environment. GuardDuty does not generate findings based on trusted IP lists. For more information, see Customizing threat detection with entity lists and IP address lists.

Threat IP list

A list of known malicious IP addresses. In addition to generating findings because of a potentially suspicious activity, GuardDuty also generates findings based on these threat lists. For more information, see Customizing threat detection with entity lists and IP address lists.

Getting started with GuardDuty

This tutorial provides a hands-on introduction to GuardDuty. The minimum requirements for enabling GuardDuty as a standalone account or as a GuardDuty administrator with AWS Organizations are covered in Step 1. Steps 2 through 5 cover using additional features recommended by GuardDuty to get the most out of your findings.

Topics

- Before you begin
- Step 1: Enable Amazon GuardDuty
- Step 2: Generate sample findings and explore basic operations
- Step 3: Configure exporting GuardDuty findings to an Amazon S3 bucket
- Step 4: Set up GuardDuty finding alerts through SNS
- Next steps

Before you begin

GuardDuty is a threat detection service that monitors Foundational data sources such as AWS CloudTrail management events, Amazon VPC Flow Logs, and Amazon Route 53 Resolver DNS query logs. GuardDuty also analyzes features associated with its protection types only if you enable them separately. Features include Kubernetes audit logs, RDS login activity, AWS CloudTrail data events for Amazon S3, Amazon EBS volumes, Runtime Monitoring, and Lambda network activity logs. Using these data sources and features (if enabled), GuardDuty generates security findings for your account.

After you enable GuardDuty, it starts monitoring your account for potential threats based on the activities in foundational data sources. By default, Extended Threat Detection is enabled for all AWS accounts that have enabled GuardDuty. This capability detects multi-stage attack sequences that span multiple foundational data sources, AWS resources, and time, in your account. To detect potential threats to specific AWS resources, you can choose to enable use-case focused protection plans that GuardDuty offers. For more information, see Features of GuardDuty.

You do not need to enable any of the foundational data sources explicitly. When you enable S3 Protection, you don't need to enable Amazon S3 data event logging explicitly. Similarly, when you enable EKS Protection, you don't need to enable Amazon EKS audit logs explicitly. Amazon GuardDuty pulls independent streams of data directly from these services.

For a new GuardDuty account, some of the available protection types that are supported in an AWS Region are enabled and included in the 30-day free trial period by default. You can opt out of any or all of them. If you've an existing AWS account with GuardDuty enabled, you can choose to enable any or all of the protection plans that are available in your Region. For an overview of protection plans and which protection plans will be enabled by default, see Pricing in GuardDuty.

When enabling GuardDuty, consider the following items:

- GuardDuty is a Regional service, meaning any of the configuration procedures you follow on this page must be repeated in each Region that you want to monitor with GuardDuty.

  We highly recommend that you enable GuardDuty in all supported AWS Regions. This enables GuardDuty to generate findings about unauthorized or unusual activity even in Regions that you are not actively using. This also enables GuardDuty to monitor AWS CloudTrail events for global AWS services such as IAM. If GuardDuty is not enabled in all supported Regions, its ability to detect activity that involves global services is reduced. For a full list of Regions where GuardDuty is available, see Regions and endpoints.
- Any user with administrator privileges in an AWS account can enable GuardDuty, however, following the security best practice of least privilege, it is recommended that you create an IAM role, user, or group to manage GuardDuty specifically. For information about the permissions required to enable GuardDuty see Permissions required to enable GuardDuty.
- When you enable GuardDuty for the first time in any AWS Region, by default, it also enables all the available protection types that are supported in that Region, including Malware Protection for EC2. GuardDuty creates a service–linked role for your account called `AWSServiceRoleForAmazonGuardDuty`. This role includes the permissions and the trust policies that allow GuardDuty to consume and analyze events directly from the GuardDuty foundational data sources to generate security findings. Malware Protection for EC2 creates another service–linked role for your account called `AWSServiceRoleForAmazonGuardDutyMalwareProtection`. This role includes the permissions and trust policies that allow Malware Protection for EC2 perform agentless scans to detect malware in your GuardDuty account. It allows GuardDuty to create an EBS volume snapshot in your account, and share that snapshot with the GuardDuty service account. For more information, see Service-linked role permissions for GuardDuty. For more information about service-linked roles, see Using service-linked roles.

- When you enable GuardDuty for the first time in any Region your AWS account is automatically enrolled in a 30-day GuardDuty free trial for that Region.

The following video explains how an administrator account can get started with GuardDuty and enable it in multiple member accounts.

Getting started: Enabling Amazon GuardDuty for standalone or multiple-account environments

Step 1: Enable Amazon GuardDuty

The first step to using GuardDuty is to enable it in your account. Once enabled, GuardDuty will immediately begin to monitor for security threats in the current Region.

If you want to manage GuardDuty findings for other accounts within your organization as a GuardDuty administrator, you must add member accounts and enable GuardDuty for them as well.

> **Note**
>
> If you want to enable GuardDuty Malware Protection for S3 without enabling GuardDuty, then for steps, see GuardDuty Malware Protection for S3.

Standalone account environment

1. Open the GuardDuty console at https://console.aws.amazon.com/guardduty/
2. Select the **Amazon GuardDuty - All features** option.
3. Choose **Get started**.
4. On the **Welcome to GuardDuty** page, view the service terms. Choose **Enable GuardDuty**.

Multi-account environment

> **Important**
>
> As prerequisites for this process, you must be in the same organization as all the accounts you want to manage, and have access to the AWS Organizations management

account in order to delegate an administrator for GuardDuty within your organization. Additional permissions may be required to delegate an administrator, for more info see Permissions required to designate a delegated GuardDuty administrator account.

To designate a delegated GuardDuty administrator account

1. Open the AWS Organizations console at https://console.aws.amazon.com/organizations/, using the management account.
2. Open the GuardDuty console at https://console.aws.amazon.com/guardduty/.

   Is GuardDuty already enabled in your account?

   - If GuardDuty is not already enabled, you can select **Get Started** and then designate a GuardDuty delegated administrator on the **Welcome to GuardDuty** page.
   - If GuardDuty is enabled, you can designate a GuardDuty delegated administrator on the **Settings** page.
3. Enter the twelve-digit AWS account ID of the account that you want to designate as the GuardDuty delegated administrator for the organization and choose **Delegate**.

   Note

   If GuardDuty is not already enabled, designating a delegated administrator will enable GuardDuty for that account in your current Region.

To add member accounts

This procedure covers adding members accounts to a GuardDuty delegated administrator account through AWS Organizations. There is also the option to add members by invitation. To learn more about both methods for associating members in GuardDuty, see Multiple accounts in Amazon GuardDuty.

1. Log in to the delegated administrator account
2. Open the GuardDuty console at https://console.aws.amazon.com/guardduty/.
3. In the navigation panel, choose **Settings**, and then choose **Accounts**.

   The accounts table displays all of the accounts in the organization.

4. Choose the accounts that you want to add as members by selecting the box next to the account ID. Then from the **Action** menu select **Add member**.

> **Tip**
>
> You can automate adding new accounts as members by turning on the **Auto-enable** feature; however, this only applies to accounts that join your organization after the feature has been enabled.

Step 2: Generate sample findings and explore basic operations

When GuardDuty discovers a security issue, it generates a finding. A GuardDuty finding is a dataset containing details relating to that unique security issue. The finding's details can be used to help you investigate the issue.

GuardDuty supports generating sample findings with placeholder values, which can be used to test GuardDuty functionality and familiarize yourself with findings before needing to respond to a real security issue discovered by GuardDuty. Follow the guide below to generate sample findings for each finding type available in GuardDuty, for additional ways to generate sample findings, including generating a simulated security event within your account, see Sample findings.

To create and explore sample findings

1. In the navigation pane, choose **Settings**.
2. On the **Settings** page, under **Sample findings**, choose **Generate sample findings**.
3. In the navigation pane, choose **Summary** to view the insights about the findings generated in your AWS environment. For more information about the components of the Summary dashboard, see Summary dashboard in Amazon GuardDuty.
4. In the navigation pane, choose **Findings**. The sample findings are displayed on the **Current findings** page with the prefix **[SAMPLE]**.
5. Select a finding from the list to display details for the finding.

    - You can review the different information fields available in the finding details pane. Different types of findings can have different fields. For more information about the available fields across all finding types see Finding details. From the details pane you can take the following actions:

- Select the **finding ID** at the top of the pane to open the complete JSON details for the finding. The complete JSON file can also be downloaded from this panel. The JSON contains some additional information not included in the console view and is the format that can be ingested by other tools and services.
- View the **Resource affected** section. In a real finding, the information here will help you identify a resource in your account that should be investigated and will include links to the appropriate AWS Management Console for actionable resources.
- Select the + or - looking glass icons to create an inclusive or exclusive filter for that detail. For more information about finding filters see Filtering findings in GuardDuty.

6. Archive all your sample findings

   a. Select all findings by selection the check box at the top of the list.

   b. Deselect any findings that you wish to keep.

   c. Select the **Actions** menu and then select **Archive** to hide the sample findings.

> **Note**
>
> To view the archived findings select **Current** and then **Archived** to switch the findings view.

Step 3: Configure exporting GuardDuty findings to an Amazon S3 bucket

GuardDuty recommends configuring settings to export findings because it allows you to export your findings to an S3 bucket for indefinite storage beyond the GuardDuty 90-day retention period. This allows you to keep records of findings or track issues within your AWS environment over time. GuardDuty encrypts the findings data in your S3 bucket by using AWS Key Management Service (AWS KMS key). To configure the settings, you must give GuardDuty the permission a KMS key. For more detailed steps, see Exporting generated findings to Amazon S3.

To export GuardDuty findings to Amazon S3 bucket

1. **Attach policy to KMS key**

   a. Sign in to the AWS Management Console and open the AWS Key Management Service (AWS KMS) console at https://console.aws.amazon.com/kms.

   b. To change the AWS Region, use the Region selector in the upper-right corner of the page.

   c. In the navigation pane, choose **Customer managed keys**.

   d. Select an existing KMS key, or perform the steps to Create a symmetric encryption KMS key in the *AWS Key Management Service Developer Guide*.

      The Region of your KMS key and Amazon S3 bucket must be the same.

      Copy the key ARN to a notepad for use in the later steps.

   e. In the **Key policy** section of your KMS key, choose **Edit**. If **Switch to policy view** is displayed, choose it to display the **Key policy**, and then choose **Edit**.

   f. Copy the following policy block to your KMS key policy:

```
{
    "Sid": "AllowGuardDutyKey",
    "Effect": "Allow",
    "Principal": {
        "Service": "guardduty.amazonaws.com"
    },
    "Action": "kms:GenerateDataKey",
    "Resource": "KMS key ARN",
    "Condition": {
        "StringEquals": {
            "aws:SourceAccount": "123456789012",
            "aws:SourceArn":
 "arn:aws:guardduty:Region2:123456789012:detector/SourceDetectorID"
        }
    }
}
```

      Edit the policy by replacing the following values that are formatted in *red* in the policy example:

1. Replace *KMS key ARN* with the Amazon Resource Name (ARN) of the KMS key. To locate the key ARN, see [Finding the key ID and ARN](#) in the *AWS Key Management Service Developer Guide*.
2. Replace *123456789012* with the AWS account ID that owns the GuardDuty account exporting the findings.
3. Replace *Region2* with the AWS Region where the GuardDuty findings are generated.
4. Replace *SourceDetectorID* with the `detectorID` of the GuardDuty account in the specific Region where the findings generated.

   To find the `detectorId` for your account and current Region, see the **Settings** page in the https://console.aws.amazon.com/guardduty/ console, or run the ListDetectors API.

2. **Attach policy to Amazon S3 bucket**

   If you do not already have an Amazon S3 bucket where you want to export these findings, see Creating a bucket in the *Amazon S3 User Guide*.

   a. Perform the steps under To create or edit a bucket policy in the *Amazon S3 User Guide*, until the **Edit bucket policy** page appears.

   b. The **example policy** shows how grant GuardDuty permission to export findings to your Amazon S3 bucket. If you change the path after you configure export findings, then you must modify the policy to grant permission to the new location.

   Copy the following **example policy** and paste it into the **Bucket policy editor**.

   If you added the policy statement before the final statement, add a comma before adding this statement. Make sure that the JSON syntax of your KMS key policy is valid.

   S3 bucket example policy

   JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "Allow GetBucketLocation",
            "Effect": "Allow",
            "Principal": {
```

```
                "Service": "guardduty.amazonaws.com"
            },
            "Action": "s3:GetBucketLocation",
            "Resource": "arn:aws:s3:::amzn-s3-demo-bucket",
            "Condition": {
                "StringEquals": {
                    "aws:SourceAccount": "123456789012",
                    "aws:SourceArn": "arn:aws:guardduty:us-
east-2:123456789012:detector/SourceDetectorID"

                }
            }
        },
        {
            "Sid": "Allow PutObject",
            "Effect": "Allow",
            "Principal": {
                "Service": "guardduty.amazonaws.com"
            },
            "Action": "s3:PutObject",
            "Resource": "arn:aws:s3:::amzn-s3-demo-bucket[optional
 prefix]/*",
            "Condition": {
                "StringEquals": {
                    "aws:SourceAccount": "123456789012",
                    "aws:SourceArn": "arn:aws:guardduty:us-
east-2:123456789012:detector/SourceDetectorID"

                }
            }
        },
        {
            "Sid": "Deny unencrypted object uploads",
            "Effect": "Deny",
            "Principal": {
                "Service": "guardduty.amazonaws.com"
            },
            "Action": "s3:PutObject",
            "Resource": "arn:aws:s3:::amzn-s3-demo-bucket[optional
 prefix]/*",
            "Condition": {
                "StringNotEquals": {
                    "s3:x-amz-server-side-encryption": "aws:kms"
                }
```

```
            }
        },
        {
            "Sid": "Deny incorrect encryption header",
            "Effect": "Deny",
            "Principal": {
                "Service": "guardduty.amazonaws.com"
            },
            "Action": "s3:PutObject",
            "Resource": "arn:aws:s3:::amzn-s3-demo-bucket[optional
 prefix]/*",
            "Condition": {
                "StringNotEquals": {
                "s3:x-amz-server-side-encryption-aws-kms-key-id":
 "arn:aws:kms:us-east-2:111122223333:key/a1b2c3d4-5678-90ab-cdef-
EXAMPLE11111"
                }
            }
        },
        {
            "Sid": "Deny non-HTTPS access",
            "Effect": "Deny",
            "Principal": "*",
            "Action": "s3:*",
            "Resource": "arn:aws:s3:::amzn-s3-demo-bucket[optional
 prefix]/*",
            "Condition": {
                "Bool": {
                    "aws:SecureTransport": "false"
                }
            }
        }
    ]
}
```

c. Edit the policy by replacing the following values that are formatted in *red* in the policy example:

1. Replace *Amazon S3 bucket ARN* with the Amazon Resource Name (ARN) of the Amazon S3 bucket. You can find the **Bucket ARN** on the **Edit bucket policy** page in the https://console.aws.amazon.com/s3/ console.

2. Replace *123456789012* with the AWS account ID that owns the GuardDuty account exporting the findings.

3. Replace *`Region2`* with the AWS Region where the GuardDuty findings are generated.
4. Replace *`SourceDetectorID`* with the `detectorID` of the GuardDuty account in the specific Region where the findings generated.

   To find the `detectorId` for your account and current Region, see the **Settings** page in the https://console.aws.amazon.com/guardduty/ console, or run the ListDetectors API.
5. Replace *`[optional prefix]`* part of the *`S3 bucket ARN/[optional prefix]`* placeholder value with an optional folder location to which you want to export the findings. For more information about the use of prefixes, see Organizing objects using prefixes in the *Amazon S3 User Guide*.

   When you provide an optional folder location that doesn't exist already, GuardDuty will create that location only if the account associated with the S3 bucket is the same as the account exporting the findings. When you export findings to an S3 bucket that belongs to another account, the folder location must exist already.
6. Replace *`KMS key ARN`* with the Amazon Resource Name (ARN) of the KMS key associated with the encryption of the findings exported to the S3 bucket. To locate the key ARN, see Finding the key ID and ARN in the *AWS Key Management Service Developer Guide*.

3. **Steps in GuardDuty console**

   a. Open the GuardDuty console at https://console.aws.amazon.com/guardduty/.

   b. In the navigation pane, choose **Settings**.

   c. On the **Settings** page, under **Findings export options,** for **S3 bucket**, choose **Configure now** (or **Edit**, as needed).

   d. For **S3 bucket ARN**, enter the **`bucket ARN`** to which you want to send the findings. To view the bucket ARN, see Viewing the properties for an S3 bucket in the *Amazon S3 User Guide*.

   e. For **KMS key ARN**, enter the **`key ARN`**. To locate the key ARN, see Find the key ID and key ARN in the *AWS Key Management Service Developer Guide*.

   f. Choose **Save**.

Step 4: Set up GuardDuty finding alerts through SNS

GuardDuty integrates with Amazon EventBridge, which can be used to send findings data to other applications and services for processing. With EventBridge you can use GuardDuty findings to initiate automatic responses to your findings by connecting finding events to targets such as AWS Lambda functions, Amazon EC2 Systems Manager automation, Amazon Simple Notification Service (SNS) and more.

In this example you will create an SNS topic to be the target of an EventBridge rule, then you'll use EventBridge to create a rule that captures findings data from GuardDuty. The resulting rule forwards the finding details to an email address. To learn how you can send findings to Slack or Amazon Chime, and also modify the types of findings alerts are sent for, see Set up an Amazon SNS topic and endpoint.

To create an SNS topic for your findings alerts

1. Open the Amazon SNS console at https://console.aws.amazon.com/sns/v3/home.
2. In the navigation pane, choose **Topics**.
3. Choose **Create Topic**.
4. For **Type**, select **Standard**.
5. For **Name**, enter **`GuardDuty`**.
6. Choose **Create Topic**. The topic details for your new topic will open.
7. In the **Subscriptions** section, choose **Create subscription**.
8. For **Protocol**, choose **Email**.
9. For **Endpoint**, enter the email address to send notifications to.
10. Choose **Create subscription**.

    After you create your subscription, you must confirm the subscription through email.
11. To check for a subscription message, go to your email inbox, and in the subscription message, choose **Confirm subscription**.

    > **Note**
    >
    > To check the email confirmation status, go to the SNS console and choose **Subscriptions**.

To create an EventBridge rule to capture GuardDuty findings and format them

1. Open the EventBridge console at https://console.aws.amazon.com/events/.
2. In the navigation pane, choose **Rules**.
3. Choose **Create rule**.
4. Enter a name and description for the rule.

   A rule can't have the same name as another rule in the same Region and on the same event bus.
5. For **Event bus**, choose **default**.
6. For **Rule type**, choose **Rule with an event pattern**.
7. Choose **Next**.
8. For **Event source**, choose **AWS events**.
9. For **Event pattern**, choose **Event pattern form**.
10. For **Event source**, choose **AWS services**.
11. For **AWS service**, choose **GuardDuty**.
12. For **Event Type**, choose **GuardDuty Finding**.
13. Choose **Next**.
14. For **Target types**, choose **AWS service**.
15. For **Select a target**, choose **SNS topic**, and for **Topic**, choose the name of the SNS topic you created earlier.
16. In the **Additional settings** section, for **Configure target input**, choose **Input transformer**.

    Adding an input transformer formats the JSON finding data sent from GuardDuty into a human-readable message.
17. Choose **Configure input transformer**.
18. In the **Target input transformer** section, for **Input path**, paste the following code:

```
{
  "severity": "$.detail.severity",
  "Finding_ID": "$.detail.id",
  "Finding_Type": "$.detail.type",
  "region": "$.region",
```

```
  "Finding_description": "$.detail.description"
}
```

19. To format the email, for **Template**, paste the following code and make sure to replace the text in red with the values appropriate to your Region:

```
"You have a severity severity GuardDuty finding type Finding_Type in
 the Region_Name Region."
"Finding Description:"
"Finding_Description."
"For more details open the GuardDuty console at https://console.aws.amazon.com/
guardduty/home?region=region#/findings?search=id%3DFinding_ID"
```

20. Choose **Confirm**.
21. Choose **Next**.
22. (Optional) Enter one or more tags for the rule. For more information, see Amazon EventBridge tags in the *Amazon EventBridge User Guide*.
23. Choose **Next**.
24. Review the details of the rule and choose **Create rule**.
25. (Optional) Test your new rule by generating sample findings with the process in Step 2. You will receive an email for each sample finding generated.

Next steps

As you continue to use GuardDuty, you will come to understand the types of findings that are relevant to your environment. Whenever you receive a new finding, you can find information, including remediation recommendations about that finding, by selecting **Learn more** from the finding description in the finding details pane, or by searching for the finding name on GuardDuty finding types.

The following features will help you tune GuardDuty so that it can provide the most relevant findings for your AWS environment:

- To easily sort findings based on specific criteria, such as instance ID, account ID, S3 bucket name, and more, you can create and save filters within GuardDuty. For more information, see Filtering findings in GuardDuty.

- If you are receiving findings for expected behavior in your environment, you can automatically archive findings based on the criteria you define with suppression rules.
- To prevent findings from being generated from a subset of trusted IPs, or to have GuardDuty monitor IPs outside it's normal monitoring scope, you can set up Trusted IP and threat lists.

GuardDuty foundational data sources

GuardDuty uses the foundational data sources to detect communication with known malicious domains and IP addresses, and identify potentially anomalous behavior and unauthorized activity. While in transit from these sources to GuardDuty, all of the log data is encrypted. GuardDuty extracts various fields from these logs sources for profiling and anomaly detection, and then discards these logs.

When you enable GuardDuty for the first time in a Region, there is a 30-day free trial that includes threat detection for all the foundational data sources. During this free trial, you can monitor an estimated monthly usage broken down by each foundational data source. As a delegated GuardDuty administrator account, you can view the estimated monthly usage cost broken down by each member account that belongs to your organization and has enabled GuardDuty. After the 30-day trial ends, you can use AWS Billing for information about the usage cost.

There is no additional cost when GuardDuty accesses the events and logs from these foundational data sources.

After you enable GuardDuty in your AWS account, it automatically starts to monitor the log sources explained in the following sections. You **don't** need to enable anything else for GuardDuty to start analyzing and processing these data sources to generate associated security findings.

Topics

- AWS CloudTrail management events
- VPC Flow Logs
- Route53 Resolver DNS query logs

AWS CloudTrail management events

AWS CloudTrail provides you with a history of AWS API calls for your account, including API calls made using the AWS Management Console, the AWS SDKs, the command line tools, and certain AWS services. CloudTrail also helps you identify which users and accounts invoked AWS APIs for services that support CloudTrail, the source IP address from where the calls were invoked, and the time at which the calls were invoked. For more information, see What is AWS CloudTrail in *AWS CloudTrail User Guide*.

GuardDuty monitors CloudTrail management events, also known as control plane events. These events provide insight into management operations that are performed on resources in your AWS account.

The following are examples of CloudTrail management events that GuardDuty monitors:

- Configuring security (IAM `AttachRolePolicy` API operations)
- Configuring rules for routing data (Amazon EC2 `CreateSubnet` API operations)
- Setting up logging (AWS CloudTrail `CreateTrail` API operations)

When you enable GuardDuty, it starts consuming CloudTrail management events directly from CloudTrail through an independent and duplicated stream of events and analyzes your CloudTrail event logs.

GuardDuty does not manage your CloudTrail events or affect your existing CloudTrail configurations. Similarly, your CloudTrail configurations don't affect how GuardDuty consumes and processes the event logs. To manage access and retention of your CloudTrail events, use the CloudTrail service console or API. For more information, see Viewing events with CloudTrail event history in *AWS CloudTrail User Guide*.

How GuardDuty handles AWS CloudTrail global events

For most AWS services, CloudTrail events are recorded in the AWS Region where they are created. For global services such as AWS Identity and Access Management (IAM), AWS Security Token Service (AWS STS), Amazon Simple Storage Service (Amazon S3), Amazon CloudFront, and Amazon Route 53 (Route 53), events are only generated in the Region where they occur but they have a global significance.

When GuardDuty consumes CloudTrail Global service events (GSE) with security value such as network configurations or user permissions, it replicates those events and processes them in each Region where you have enabled GuardDuty. This behavior helps GuardDuty maintain user and role profiles in each Region, which is vital to detecting anomalous events.

> **Note**
>
> For findings generated from these global service events, the Region value in the finding may differ from the Region where GuardDuty creates the detection. For example, a finding

might show `us-east-1` as the Region even if GuardDuty creates the detection in a different Region.

We recommend that you enable GuardDuty in all AWS Regions available in your AWS account. Even if you don't have resources deployed in certain Regions, enabling GuardDuty helps protect your account from potential threats. Threat actors can potentially launch attacks through global services (such as IAM, AWS STS, or Amazon CloudFront). They can attempt to create unauthorized resources to exploit Regions where you have limited presence. GuardDuty processes global service events in all Regions where you've enabled the service, including both default and opt-in Regions. This helps GuardDuty detect potentially suspicious activities across your AWS account, including Regions where you don't actively use resources.

VPC Flow Logs

The VPC Flow Logs feature of Amazon VPC captures information about the IP traffic going to and from network interfaces attached to the Amazon Elastic Compute Cloud (Amazon EC2) instances within your AWS environment.

When you enable GuardDuty, it immediately starts analyzing your VPC flow logs from Amazon EC2 instances within your account. It consumes VPC flow log events directly from the VPC Flow Logs feature through an independent and duplicate stream of flow logs. This process does not affect any of your existing flow logs configuration.

Lambda Protection

Lambda Protection is an optional enhancement to Amazon GuardDuty. Presently, Lambda Network Activity Monitoring includes Amazon VPC flow logs from all Lambda functions for your account, even those logs that don't use VPC networking. To protect your Lambda function from potential security threats, you will need to configure Lambda Protection in your GuardDuty account. For more information, see Lambda Protection.

GuardDuty Runtime Monitoring

When you manage the security agent (either manually or through GuardDuty) in EKS Runtime Monitoring or Runtime Monitoring for EC2 instances, and GuardDuty is presently deployed on an Amazon EC2 instance and receives the Collected runtime event types from this instance, GuardDuty will not charge your AWS account for the analysis of VPC flow logs from this Amazon EC2 instance. This helps GuardDuty avoid double usage cost in the account.

GuardDuty doesn't manage your flow logs or make them accessible in your account. To manage access to and retention of your flow logs, you must configure the VPC Flow Logs feature.

Route53 Resolver DNS query logs

If you use AWS DNS resolvers for your Amazon EC2 instances (the default setting), then GuardDuty can access and process your request and response Route53 Resolver DNS query logs through the internal AWS DNS resolvers. If you use another DNS resolver, such as OpenDNS or GoogleDNS, or if you set up your own DNS resolvers, then GuardDuty cannot access and process data from this data source.

When you enable GuardDuty, it immediately starts analyzing your Route53 Resolver DNS query logs from an independent stream of data. This data stream is separate from the data provided through the Route 53 Resolver query logging feature. Configuration of this feature does not affect GuardDuty analysis.

> **Note**
>
> GuardDuty doesn't support monitoring DNS logs for Amazon EC2 instances that are launched on AWS Outposts because the Amazon Route 53 Resolver query logging feature is not available in that environment.

GuardDuty Extended Threat Detection

GuardDuty Extended Threat Detection automatically detects multi-stage attacks that span data sources, multiple types of AWS resources, and time, within an AWS account. With this capability, GuardDuty focuses on the sequence of multiple events that it observes by monitoring different types of data sources. Extended Threat Detection correlates these events to identify scenarios that present themselves as a potential threat to your AWS environment, and then generates an attack sequence finding.

Topics

- Attack sequence threat scenario examples
- How it works
- Enabling protection plans to maximize threat detection
- Extended Threat Detection in GuardDuty console
- Understanding and managing attack sequence findings
- Additional resources

Attack sequence threat scenario examples

Extended Threat Detection covers threat scenarios that involve compromise related to AWS credentials misuse, data compromise attempts in Amazon S3 buckets, and container and Kubernetes resource compromise in Amazon EKS clusters. A single finding can encompass an entire attack sequence. For example, the following list describes the scenarios that GuardDuty might detect:

Example 1 - AWS credentials and Amazon S3 bucket data compromise

- A threat actor gaining unauthorized access to a compute workload.
- The actor then performing a series of actions such as privilege escalation and establishing persistence.
- Finally, the actor exfiltrating data from an Amazon S3 resource.

Example 2 - Amazon EKS cluster compromise

- A threat actor attempts to exploit a container application within an Amazon EKS cluster.
- The actor uses that compromised container to obtain privileged service account tokens.

- The actor then leverages these elevated privileges to access sensitive Kubernetes secrets or AWS resources through pod identities.

Because of the nature of the associated threat scenarios, GuardDuty considers all Attack sequence finding types as **Critical**.

The following video provides a demonstration of how you can use Extended Threat Detection.

Amazon GuardDuty Extended Threat Detection demonstration

How it works

When you enable Amazon GuardDuty in your account in a specific AWS Region, Extended Threat Detection is also enabled by default. There is no additional cost associated with the usage of Extended Threat Detection. By default, it correlates events across all Foundational data sources. However, when you enable more GuardDuty protection plans, such as S3 Protection, EKS Protection, and Runtime Monitoring, this will open additional types of attack sequence detections by widening the range of event sources. This will potentially help with a more comprehensive threat analysis and better detection of attack sequences. For more information, see Enabling protection plans to maximize threat detection.

GuardDuty correlates multiple events, including API activities and GuardDuty findings. These events are called **Signals**. Sometimes, there might be events in your environment that, on their own, don't present themselves as a clear potential threat. GuardDuty terms them as *weak* signals. With Extended Threat Detection, GuardDuty identifies when a sequence of multiple actions align to a potentially suspicious activity, and generates an attack sequence finding in your account. These multiple actions can include weak signals and already identified GuardDuty findings in your account.

> **Note**
>
> When correlating events for attack sequences, Extended Threat Detection doesn't consider archived findings, including those findings that are automatically archived because of Suppression rules. This behavior ensures that only active, relevant signals contribute to attack sequence detection. To ensure that you're not impacted by this, review existing suppression rules in your account. For more information, see Using suppression rules with Extended Threat Detection.

GuardDuty is also designed to identify potential in-progress or recent attack behaviors (within a 24-hour rolling time window) in your account. For example, an attack could start by an actor gaining unintended access to a compute workload. The actor would then perform a series of steps, including enumeration, escalation of privileges, and exfiltration of AWS credentials. These credentials could potentially be used for further compromise or malicious access to data.

Enabling protection plans to maximize threat detection

For any GuardDuty account in a Region, the Extended Threat Detection capability gets enabled automatically. By default, this capability takes into consideration the multiple events across all Foundational data sources. To benefit from this capability, you don't need to enable all the use-case focused GuardDuty protection plans. For example, with foundational threat detection, GuardDuty can identify a potential attack sequence starting from IAM privilege discovery activity on Amazon S3 APIs, and detect subsequent S3 control plane alterations, such as changes that make bucket resource policy more permissive.

Extended Threat Detection is designed in a way that if you enable more protection plans, it helps GuardDuty correlate more diverse signals across multiple data sources. This will potentially enhance the breadth of security signals for comprehensive threat analysis and coverage of attack sequences. To identify findings that could potentially be one of the multiple stages in an attack sequence, GuardDuty **recommends** enabling specific protection plans – S3 Protection, EKS Protection, and Runtime Monitoring (with EKS add-on).

Topics

- Detecting attack sequences in Amazon EKS clusters
- Detecting attack sequences in Amazon S3 buckets

Detecting attack sequences in Amazon EKS clusters

GuardDuty correlated multiple security signals across EKS audit logs, runtime behavior of processes, and AWS API activity to detect sophisticated attack patterns. To benefit from Extended Threat Detection for EKS, you must enable at least one of these features – EKS Protection or Runtime Monitoring (with EKS add-on). EKS Protection monitors control plane activities through audit logs, while Runtime Monitoring observes behaviors within containers.

For maximum coverage and comprehensive threat detection, GuardDuty recommends enabling both protection plans. Together, they create a complete view of your EKS clusters, enabling

GuardDuty to detect complex attack patterns. For example, it can identify an anomalous deployment of a privileged container (detected with EKS Protection), followed by persistence attempts, crypto-mining, and reverse shell creation within that container (detected with Runtime Monitoring). GuardDuty represents these related events as a single, critical-severity finding, called AttackSequence:EKS/CompromisedCluster. When you enable both the protection plans, the attack sequence finding covers the following threat scenarios:

- Compromise of containers running vulnerable web applications
- Unauthorized access through misconfigured credentials
- Attempts to escalate privileges
- Suspicious API requests
- Attempts to access data maliciously

The following list provides details when these dedicated protection plans are enabled individually:

EKS Protection

Enabling EKS Protection gives GuardDuty an ability to detect attack sequences involving Amazon EKS cluster control plane activities. This allows GuardDuty to correlate EKS audit logs and AWS API activity. For example, GuardDuty can detect an attack sequence where an actor attempts unauthorized access to cluster secrets, modifies Kubernetes role-based access control (RBAC) permissions, and creates privileged pods. For more information about enabling this protection plan, see EKS Protection.

Runtime Monitoring for Amazon EKS

Enabling Runtime Monitoring for Amazon EKS clusters gives GuardDuty an ability to enhance EKS attack sequence detection with container-level visibility. This helps GuardDuty detect potential malicious processes, suspicious runtime behaviors, and potential malware execution. For example, GuardDuty can detect an attack sequence where a container starts exhibiting suspicious behavior, such as cryptomining processes or establishing connections to known malicious endpoints. For more information about enabling this protection plan, see Runtime Monitoring.

If you don't enable EKS Protection or Runtime Monitoring, GuardDuty will not be able to generate individual EKS Protection finding types or Runtime Monitoring finding types. Therefore, GuardDuty will not be able to detect multi-stage attack sequences that involve associated findings.

Detecting attack sequences in Amazon S3 buckets

Enabling S3 Protection gives GuardDuty an ability to detect attack sequences involving attempts to data compromise in your Amazon S3 buckets. Without S3 Protection, GuardDuty can detect when your S3 bucket resource policy becomes overly permissive. When you enable S3 Protection, GuardDuty gains the ability to detect potential data exfiltration activities that may occur after your S3 bucket becomes overly permissive.

If S3 Protection is not enabled, GuardDuty will not be able to generate individual S3 Protection finding types. Therefore, GuardDuty will not be able to detect multi-stage attack sequences that involve associated findings. For more information about enabling this protection plan, see S3 Protection.

Extended Threat Detection in GuardDuty console

By default, the Extended Threat Detection page in GuardDuty console displays the **Status** as **Enabled**. With foundational threat detection, the status represents that GuardDuty can detect a potential attack sequence involving IAM privilege discovery activity on Amazon S3 APIs and detecting subsequent S3 control plane alterations.

Use the following steps to access the Extended Threat Detection page in GuardDuty console:

1. You can open GuardDuty console at https://console.aws.amazon.com/guardduty/.
2. In the left navigation pane, choose **Extended Threat Detection**.

   This page provides details about the threat scenarios that Extended Threat Detection covers.
3. On the **Extended Threat Detection** page, view the **Related protection plans** section. If you **want to** enable dedicated protection plans to enhance threat detection coverage in your account, select **Configure** option for that protection plan.

Understanding and managing attack sequence findings

Attack sequence findings are just like other GuardDuty findings in your account. You can view them on the **Findings** page in the GuardDuty console. For information about viewing findings, see Findings page in GuardDuty console.

Similar to other GuardDuty findings, attack sequence findings are also automatically sent to Amazon EventBridge. Based on your settings, attack sequence findings are also exported to a publishing destination (Amazon S3 bucket). To set a new publishing destination or update an existing one, see Exporting generated findings to Amazon S3.

Additional resources

View the following sections to gain more understanding about attack sequences:

- After learning about Extended Threat Detection and attack sequences, you can generate sample attack sequence finding types by following the steps in Sample findings.
- Learn about Attack sequence finding types.
- Review findings and explore finding details associated with Attack sequence finding details.
- Prioritize and address attack sequence finding types by following the steps for the associated impacted resources in Remediating findings.

GuardDuty EKS Protection

EKS Protection helps you detect potential security risks in Amazon Elastic Kubernetes Service (Amazon EKS) clusters in your AWS environment. For example, it helps you detect when a misconfigured EKS cluster is being accessed by an unauthenticated actor that attempts to collect secrets or AWS credentials from your cluster. EKS Protection uses EKS audit logs to analyze activities of users and applications.

When you enable EKS Protection, GuardDuty automatically starts monitoring your Amazon EKS clusters for potential security threats. GuardDuty uses its own independent stream to collect and analyze EKS audit logs in EKS Protection – no additional configuration is required.

When GuardDuty detects a potential threat based on EKS audit log monitoring, it generates a security finding. For information about the finding types that GuardDuty may generate when you enable EKS Protection, see EKS Protection finding types.

> **Note**
>
> To view EKS audit logs in your account (optional), you can configure Amazon EKS control plane logging to send audit logs to CloudWatch Logs. This configuration is separate from EKS Protection and is not required for security monitoring capability in GuardDuty.

30-day free trial

- When you enable GuardDuty in an AWS account in an AWS Region for the first time, you get a 30-day free trial. In this case, GuardDuty will also enable EKS Protection, which is included in the 30-day free trial.
- When you are already using GuardDuty and decide to enable EKS Protection for the first time, your account in this Region will get a 30-day free trial for EKS Protection.
- You can choose to disable EKS Protection in any Region at any time.
- During the 30-day free trial, you can get an estimate of your usage costs in that account and Region. After the 30-day free trial ends, GuardDuty doesn't automatically disable EKS Protection. Your account in this Region will start incurring usage cost. For more information, see Estimating usage cost.

When you disable EKS Protection, GuardDuty immediately stops monitoring and analyzing the EKS audit logs for your Amazon EKS resources.

EKS Protection may not be available in all the AWS Regions where GuardDuty is available. For more information, see Region-specific feature availability.

> **Note**
>
> EKS Runtime Monitoring is managed as a part of Runtime Monitoring. For more information, see GuardDuty Runtime Monitoring.

EKS audit logs in EKS Protection

EKS audit logs capture sequential actions within your Amazon EKS cluster, including activities from users, applications using the Kubernetes API, and the control plane. Audit logging is a component of all Kubernetes clusters.

For more information, see Auditing in the Kubernetes documentation.

Amazon EKS allows EKS audit logs to be ingested as Amazon CloudWatch Logs through the EKS control plane logging feature. GuardDuty doesn't manage your Amazon EKS control plane logging or make EKS audit logs accessible in your account if you have not enabled them for Amazon EKS. To manage access to and retention of your EKS audit logs, you must configure the Amazon EKS control plane logging feature. For more information, see Enabling and disabling control plane logs in the Amazon EKS User Guide.

Enabling EKS Protection in multiple-account environments

In a multiple-account environment, only the delegated GuardDuty administrator account has the option to enable or disable the EKS Protection; feature for the member accounts in their organization. The GuardDuty member accounts can't modify this configuration from their accounts. The delegated GuardDuty administrator account manages their member accounts using AWS Organizations. This delegated GuardDuty administrator account can choose to auto-enable EKS Protection for all the new accounts as they join the organization. For more information about multiple-account environments, see Managing multiple accounts in Amazon GuardDuty.

Configuring EKS Audit Log Monitoring for delegated GuardDuty administrator account

Choose your preferred access method to configure EKS Audit Log Monitoring for the delegated GuardDuty administrator account.

Console

1. Open the GuardDuty console at https://console.aws.amazon.com/guardduty/.
2. In the navigation pane, choose EKS Protection.
3. Under the **Configuration** tab, you can view the current configuration status of EKS Audit Log Monitoring in the respective section. To update the configuration for delegated GuardDuty administrator account, choose **Edit** in the **EKS Audit Log Monitoring** pane.
4. Do one of the following:

   Using Enable for all accounts

   - Choose **Enable for all accounts**. This will enable the protection plan for all the active GuardDuty accounts in your AWS organization, including the new accounts that join the organization.
   - Choose **Save**.

   Using Configure accounts manually

   - To enable the protection plan only for the delegated GuardDuty administrator account account, choose **Configure accounts manually**.
   - Choose **Enable** under the **delegated GuardDuty administrator account (this account)** section.
   - Choose **Save**.

API/CLI

Run the [updateDetector](https://docs.aws.amazon.com/guardduty/latest/APIReference/API_UpdateDetector.html) API operation using your own regional detector ID and passing the `features` object name as `EKS_AUDIT_LOGS` and `status` as `ENABLED` or `DISABLED`.

To find the `detectorId` for your account and current Region, see the **Settings** page in the https://console.aws.amazon.com/guardduty/ console, or run the ListDetectors API.

You can enable or disable EKS Audit Log Monitoring by running the following AWS CLI command. Make sure to use delegated GuardDuty administrator account's valid *detector ID*.

Note

The following example code enables EKS Audit Log Monitoring. Make sure to replace *12abc34d567e8fa901bc2d34e56789f0* with the `detector-id` of the delegated GuardDuty administrator account and *555555555555* with the AWS account of the delegated GuardDuty administrator account.

To find the `detectorId` for your account and current Region, see the **Settings** page in the https://console.aws.amazon.com/guardduty/ console, or run the ListDetectors API.

```
aws guardduty update-detector --detector-id 12abc34d567e8fa901bc2d34e56789f0 --
features '[{"Name": "EKS_AUDIT_LOGS", "Status": "ENABLED"}]'
```

To disable EKS Audit Log Monitoring, replace `ENABLED` with `DISABLED`.

Auto-enable EKS Audit Log Monitoring for all member accounts

Choose your preferred access method to enable the EKS Audit Log Monitoring for existing member accounts in your organization.

Console

1. Sign in to the AWS Management Console and open the GuardDuty console at https://console.aws.amazon.com/guardduty/.

   Make sure to use the delegated GuardDuty administrator account credentials.

2. Do one of the following:

   Using the EKS Protection page

   1. In the navigation pane, choose **EKS Protection**.
   2. Under the **Configuration** tab, you can view the current status of EKS Audit Log Monitoring for active member accounts in your organization.

      To update the EKS Audit Log Monitoring configuration, choose **Edit**.

3. Choose **Enable for all accounts**. This action automatically enables EKS Audit Log Monitoring for both the existing and new accounts in the organization.
4. Choose **Save**.

> **Note**
>
> It may take up to 24 hours to update the configuration for the member accounts.

Using the Accounts page

1. In the navigation pane, choose **Accounts**.
2. On the **Accounts** page, choose **Auto-enable** preferences before **Add accounts by invitation**.
3. In the **Manage auto-enable preferences** window, choose **Enable for all accounts** under **EKS Audit Log Monitoring**.
4. Choose **Save**.

If you can't use the **Enable for all accounts** option and want to customize EKS Audit Log Monitoring configuration for specific accounts in your organization, see Selectively enable or disable EKS Audit Log Monitoring for member accounts.

API/CLI

- To selectively enable or disable EKS Audit Log Monitoring for your member accounts, run the updateMemberDetectors API operation using your own *detector ID*.
- The following example shows how you can enable EKS Audit Log Monitoring for a single member account. To disable it, replace `ENABLED` with `DISABLED`.

  To find the `detectorId` for your account and current Region, see the **Settings** page in the https://console.aws.amazon.com/guardduty/ console, or run the ListDetectors API.

  ```
  aws guardduty update-member-detectors --detector-
  id 12abc34d567e8fa901bc2d34e56789f0 --account-ids 111122223333 --features
   '[{"name": "EKS_AUDIT_LOGS", "status": "ENABLED"}]'
  ```

> **Note**
>
> You can also pass a list of account IDs separated by a space.

- When the code has successfully executed, it returns an empty list of `UnprocessedAccounts`. If there were any problems changing the detector settings for an account, that account ID is listed along with a summary of the issue.

Enable EKS Audit Log Monitoring for all existing active member accounts

Choose your preferred access method to enable EKS Audit Log Monitoring for all existing active member accounts in the organization.

Console

1. Sign in to the AWS Management Console and open the GuardDuty console at https://console.aws.amazon.com/guardduty/.

   Sign in using the delegated GuardDuty administrator account credentials.
2. In the navigation pane, choose **EKS Protection**.
3. On the **EKS Protection** page, you can view the current status of the **GuardDuty-initiated malware scan** configuration. Under the **Active member accounts** section, choose **Actions**.
4. From the **Actions** dropdown menu, choose **Enable for all existing active member accounts**.
5. Choose **Save**.

API/CLI

- To selectively enable or disable EKS Audit Log Monitoring for your member accounts, run the updateMemberDetectors API operation using your own *`detector ID`*.
- The following example shows how you can enable EKS Audit Log Monitoring for a single member account. To disable it, replace `ENABLED` with `DISABLED`.

  To find the `detectorId` for your account and current Region, see the **Settings** page in the https://console.aws.amazon.com/guardduty/ console, or run the ListDetectors API.

```
aws guardduty update-member-detectors --detector-
id 12abc34d567e8fa901bc2d34e56789f0 --account-ids 111122223333 --features
 '[{"name": "EKS_AUDIT_LOGS", "status": "ENABLED"}]'
```

> **Note**
>
> You can also pass a list of account IDs separated by a space.

- When the code has successfully executed, it returns an empty list of `UnprocessedAccounts`. If there were any problems changing the detector settings for an account, that account ID is listed along with a summary of the issue.

Auto-enable EKS Audit Log Monitoring for new member accounts

The newly added member accounts must **Enable** GuardDuty before selecting configuring GuardDuty-initiated malware scan. The member accounts managed by invitation can configure GuardDuty-initiated malware scan manually for their accounts. For more information, see Step 3 - Accept an invitation.

Choose your preferred access method to enable EKS Audit Log Monitoring for new accounts that join your organization.

Console

The delegated GuardDuty administrator account can enable EKS Audit Log Monitoring for new member accounts in an organization, using either the **EKS Audit Log Monitoring** or **Accounts** page.

To auto-enable EKS Audit Log Monitoring for new member accounts

1. Open the GuardDuty console at https://console.aws.amazon.com/guardduty/.

   Make sure to use the delegated GuardDuty administrator account credentials.
2. Do one of the following:
   - Using the **EKS Protection** page:
     1. In the navigation pane, choose **EKS Protection**.
     2. On the **EKS Protection** page, choose **Edit** in the **EKS Audit Log Monitoring**.

3. Choose **Configure accounts manually**.
4. Select **Automatically enable for new member accounts**. This step ensures that whenever a new account joins your organization, EKS Audit Log Monitoring will be automatically enabled for their account. Only the organization delegated GuardDuty administrator account can modify this configuration.
5. Choose **Save**.

- Using the **Accounts** page:

1. In the navigation pane, choose **Accounts**.
2. On the **Accounts** page, choose **Auto-enable** preferences.
3. In the **Manage auto-enable preferences** window, select **Enable for new accounts** under **EKS Audit Log Monitoring**.
4. Choose **Save**.

API/CLI

- To selectively enable or disable EKS Audit Log Monitoring for your new accounts, run the UpdateOrganizationConfiguration API operation using your own *detector ID*.
- The following example shows how you can enable EKS Audit Log Monitoring for the new members that join your organization. You can also pass a list of account IDs separated by a space.

  To find the `detectorId` for your account and current Region, see the **Settings** page in the https://console.aws.amazon.com/guardduty/ console, or run the ListDetectors API.

```
aws guardduty update-organization-configuration --detector-
id 12abc34d567e8fa901bc2d34e56789f0 --auto-enable --features '[{"Name":
 "EKS_AUDIT_LOGS", "AutoEnable": "NEW"}]'
```

Selectively enable or disable EKS Audit Log Monitoring for member accounts

Choose your preferred access method to enable or disable EKS Audit Log Monitoring for selective member accounts in your organization.

Console

1. Open the GuardDuty console at https://console.aws.amazon.com/guardduty/.

   Make sure to use the delegated GuardDuty administrator account credentials.
2. In the navigation pane, choose **Accounts**.

   On the **Accounts** page, review the **EKS Audit Log Monitoring** column for the status of your member account.
3. **To enable or disable EKS Audit Log Monitoring**

   Select an account that you want to configure for EKS Audit Log Monitoring. You can select multiple accounts at a time. Under the **Edit Protection Plans** dropdown, choose **EKS Audit Log Monitoring**, and then choose the appropriate option.

API/CLI

To selectively enable or disable EKS Audit Log Monitoring for your member accounts, invoke the updateMemberDetectors API operation using your own *`detector ID`*.

The following example shows how you can enable EKS Audit Log Monitoring for a single member account. To disable it, replace `ENABLED` with `DISABLED`. You can also pass a list of account IDs separated by a space.

To find the `detectorId` for your account and current Region, see the **Settings** page in the https://console.aws.amazon.com/guardduty/ console, or run the ListDetectors API.

```
aws guardduty update-member-detectors --detector-id 12abc34d567e8fa901bc2d34e56789f0
 --accountids 111122223333 --features '[{"Name": "EKS_AUDIT_LOGS", "Status":
 "ENABLED"}]'
```

Enabling EKS Protection for a standalone account

A standalone account owns the decision to enable or disable a protection plan in their AWS account in a specific Region.

If your account is associated with a GuardDuty administrator account through AWS Organizations, or by the method of invitation, this section doesn't apply to you. For information about managing multiple accounts, see Enabling EKS Protection in multiple-account environments.

After you enable EKS Protection, GuardDuty will start monitoring EKS audit logs for the Amazon EKS clusters in your account.

Choose your preferred access method to enable EKS Protection in your standalone account.

Console

1. Open the GuardDuty console at https://console.aws.amazon.com/guardduty/.
2. From the **Region** selector in the upper-right corner, select a Region where you want to enable EKS Protection.
3. In the navigation pane, choose EKS Protection.
4. The **EKS Protection** page provides the current status of EKS Protection for your account. Choose **Enable** to enable EKS Protection.
5. Choose **Confirm** to save your selection.

API/CLI

- Run the updateDetector API operation using the regional detector ID of the delegated GuardDuty administrator account and passing the `features` object name as `EKS_AUDIT_LOGS` and status as `ENABLED`.

  Alternatively, you can also enable EKS Protection running the a AWS CLI command. Run the following command, and replace *`12abc34d567e8fa901bc2d34e56789f0`* with your account's detector ID and *`us-east-1`* with the Region where you want to enable EKS Protection.

  To find the `detectorId` for your account and current Region, see the **Settings** page in the https://console.aws.amazon.com/guardduty/ console, or run the ListDetectors API.

  ```
  aws guardduty update-detector --detector-id 12abc34d567e8fa901bc2d34e56789f0 --
  region us-east-1 --features [{"Name" : "EKS_AUDIT_LOGS", "Status" : "ENABLED"}]'
  ```

GuardDuty S3 Protection

S3 Protection helps you detect potential security risks for data, such as data exfiltration and destruction, in your Amazon Simple Storage Service (Amazon S3) buckets. GuardDuty monitors AWS CloudTrail data events for Amazon S3, that includes object-level API operations to identify these risks in all the Amazon S3 buckets in your account.

When GuardDuty detects a potential threat based on S3 data event monitoring, it generates a security finding. For information about the finding types that GuardDuty may generate when you enable S3 Protection, see GuardDuty S3 Protection finding types.

By default, foundational threat detection includes monitoring AWS CloudTrail management events to identify potential threats in your Amazon S3 resources. This data source is different from the AWS CloudTrail data events for S3 as they both monitor different kinds of activities in your environment.

You can enable S3 Protection in an account in any Region where GuardDuty supports this feature. This will help you monitor CloudTrail data events for S3 in that account and Region. After you enable S3 Protection, GuardDuty will be able to fully monitor your Amazon S3 buckets and generate findings for suspicious access to the data stored in your S3 buckets.

To use S3 Protection, you don't need to explicitly enable or configure S3 data event logging in AWS CloudTrail.

30-day free trial

The following list explains how the 30-day free trial would work for your account:

- When you enable GuardDuty in an AWS account in a new Region for the first time, you get a 30-day free trial. In this case, GuardDuty will also enable S3 Protection, which is included in the free trial.
- When you are already using GuardDuty and decide to enable S3 Protection for the first time, your account in this Region will get a 30-day free trial for S3 Protection.
- You can choose to disable S3 Protection in any Region at any time.
- During the 30-day free trial, you can get an estimate of your usage costs in that account and Region. After the 30-day free trial ends, S3 Protection doesn't get disabled automatically. Your account in this Region will start incurring usage cost. For more information, see Estimating GuardDuty usage cost.

AWS CloudTrail data events for S3

Data events, also known as data plane operations, provide insight into the resource operations performed on or within a resource. They are often high-volume activities.

The following are examples of CloudTrail data events for S3 that GuardDuty can monitor:

- `GetObject` API operations
- `PutObject` API operations
- `ListObjects` API operations
- `DeleteObject` API operations

For more information about these APIs, see Amazon Simple Storage Service API Reference.

How GuardDuty uses CloudTrail data events for S3

When you enable S3 Protection, GuardDuty begins to analyze CloudTrail data events for S3 from all of your S3 buckets, and monitors them for malicious and suspicious activity. For more information, see AWS CloudTrail management events.

When an unauthenticated user accesses an S3 object, it means that the S3 object is publicly accessible. Therefore, GuardDuty doesn't process such requests. GuardDuty processes the requests made to the S3 objects by using valid IAM (AWS Identity and Access Management) or AWS STS (AWS Security Token Service) credentials.

> **Note**
>
> After enabling S3 Protection, GuardDuty monitors the data events from those Amazon S3 buckets that reside in the same Region where you enabled GuardDuty.

If you disable S3 Protection in your account in a specific Region, GuardDuty stops S3 data event monitoring of the data stored in your S3 buckets. GuardDuty will no longer generate S3 Protection finding types for your account in that Region.

GuardDuty using CloudTrail data events for S3 for attack sequences

GuardDuty Extended Threat Detection detects multi-stage attack sequences that span foundational data sources, AWS resources, and timeline, in an account. When GuardDuty observes a sequence of events that is indicative of a recent or an in-progress suspicious activity in your account, GuardDuty generates associated attack sequence finding.

By default, when you enable GuardDuty, Extended Threat Detection also gets enabled in your account. This capability covers the threat scenario associated with CloudTrail management events at no additional cost. However, to use Extended Threat Detection at its full potential, GuardDuty recommends enabling S3 Protection to cover threat scenarios associated with CloudTrail data events for S3.

After you enable S3 Protection, GuardDuty will automatically cover the attack sequence threat scenarios, such as compromise or destruction of data, where your Amazon S3 resources might be involved.

Enabling S3 Protection in multiple-account environments

In a multi-account environment, only the delegated GuardDuty administrator account has the option to configure (enable or disable) S3 Protection for the member accounts in their AWS organization. The GuardDuty member accounts can't modify this configuration from their accounts. The delegated GuardDuty administrator account manages their member accounts using AWS Organizations. The delegated GuardDuty administrator account can choose to have S3 Protection automatically enabled on all accounts, only new accounts, or no accounts in the organization. For more information, see Managing accounts with AWS Organizations.

Enabling S3 Protection for delegated GuardDuty administrator account

Choose your preferred access method to enable S3 Protection for the delegated GuardDuty administrator account.

Console

1. Open the GuardDuty console at https://console.aws.amazon.com/guardduty/.
2. In the navigation pane, choose **S3 Protection**.
3. On the **S3 Protection** page, choose **Edit**.
4. Do one of the following:

Using Enable for all accounts

- Choose **Enable for all accounts**. This will enable the protection plan for all the active GuardDuty accounts in your AWS organization, including the new accounts that join the organization.
- Choose **Save**.

Using Configure accounts manually

- To enable the protection plan only for the delegated GuardDuty administrator account account, choose **Configure accounts manually**.
- Choose **Enable** under the **delegated GuardDuty administrator account (this account)** section.
- Choose **Save**.

API/CLI

Run updateDetector by using the detector ID of the delegated GuardDuty administrator account for the current Region and passing the `features` object `name` as `S3_DATA_EVENTS` and `status` as `ENABLED`.

Alternatively, you can configure S3 Protection by using AWS Command Line Interface. Run the following command, and make sure to replace *12abc34d567e8fa901bc2d34e56789f0* with the detector ID of the delegated GuardDuty administrator account for the current Region.

To find the `detectorId` for your account and current Region, see the **Settings** page in the https://console.aws.amazon.com/guardduty/ console, or run the ListDetectors API.

```
aws guardduty update-detector --detector-id 12abc34d567e8fa901bc2d34e56789f0 --
features '[{"Name": "S3_DATA_EVENTS", "Status": "ENABLED"}]'
```

Auto-enable S3 Protection for all member accounts in the organization

Choose your preferred access method to enable S3 Protection for the delegated GuardDuty administrator account.

Console

1. Open the GuardDuty console at https://console.aws.amazon.com/guardduty/.

   Sign in using your administrator account account.

2. Do one of the following:

   Using the S3 Protection page

   1. In the navigation pane, choose **S3 Protection**.
   2. Choose **Enable for all accounts**. This action automatically enables S3 Protection for both existing and new accounts in the organization.
   3. Choose **Save**.

   > **Note**
   >
   > It may take up to 24 hours to update the configuration for the member accounts.

   Using the Accounts page

   1. In the navigation pane, choose **Accounts**.
   2. On the **Accounts** page, choose **Auto-enable** preferences before **Add accounts by invitation**.
   3. In the **Manage auto-enable preferences** window, choose **Enable for all accounts** under **S3 Protection**.
   4. Choose **Save**.

   If you can't use the **Enable for all accounts** option, see Selectively enable S3 Protection in member accounts.

API/CLI

- To selectively enable S3 Protection for your member accounts, invoke the updateMemberDetectors API operation using your own *detector ID*.

- The following example shows how you can enable S3 Protection for a single member account. Make sure to replace *12abc34d567e8fa901bc2d34e56789f0* with the detector-id of the delegated GuardDuty administrator account, and *111122223333*.

  To find the detectorId for your account and current Region, see the **Settings** page in the https://console.aws.amazon.com/guardduty/ console, or run the ListDetectors API.

  ```
  aws guardduty update-member-detectors --detector-
  id 12abc34d567e8fa901bc2d34e56789f0 --account-ids 111122223333 --features
   '[{"name": "S3_DATA_EVENTS", "status": "ENABLED"}]'
  ```

  > **Note**
  >
  > You can also pass a list of account IDs separated by a space.

- When the code has successfully executed, it returns an empty list of UnprocessedAccounts. If there were any problems changing the detector settings for an account, that account ID is listed along with a summary of the issue.

Enable S3 Protection for all existing active member accounts

Choose your preferred access method to enable S3 Protection for all the existing active member accounts in your organization.

Console

1. Sign in to the AWS Management Console and open the GuardDuty console at https://console.aws.amazon.com/guardduty/.

   Sign in using the delegated GuardDuty administrator account credentials.

2. In the navigation pane, choose **S3 Protection**.
3. On the **S3 Protection** page, you can view the current status of the configuration. Under the **Active member accounts** section, choose **Actions**.
4. From the **Actions** dropdown menu, choose **Enable for all existing active member accounts**.
5. Choose **Confirm**.

API/CLI

- To selectively enable S3 Protection for your member accounts, invoke the [updateMemberDetectors](https://docs.aws.amazon.com/guardduty/latest/APIReference/API_UpdateMemberDetectors.html) API operation using your own *detector ID*.
- The following example shows how you can enable S3 Protection for a single member account. Make sure to replace *12abc34d567e8fa901bc2d34e56789f0* with the `detector-id` of the delegated GuardDuty administrator account, and *111122223333*.

  To find the `detectorId` for your account and current Region, see the **Settings** page in the https://console.aws.amazon.com/guardduty/ console, or run the ListDetectors API.

  ```
  aws guardduty update-member-detectors --detector-
  id 12abc34d567e8fa901bc2d34e56789f0 --account-ids 111122223333 --features
   '[{"name": "S3_DATA_EVENTS", "status": "ENABLED"}]'
  ```

  > **Note**
  >
  > You can also pass a list of account IDs separated by a space.

- When the code has successfully executed, it returns an empty list of `UnprocessedAccounts`. If there were any problems changing the detector settings for an account, that account ID is listed along with a summary of the issue.

Auto-enable S3 Protection for new member accounts

Choose your preferred access method to enable S3 Protection for new accounts that join your organization.

Console

The delegated GuardDuty administrator account can enable for new member accounts in an organization through the console, using either the **S3 Protection** or **Accounts** page.

To auto-enable S3 Protection for new member accounts

1. Open the GuardDuty console at https://console.aws.amazon.com/guardduty/.

   Make sure to use the delegated GuardDuty administrator account credentials.
2. Do one of the following:

- Using the **S3 Protection** page:

  1. In the navigation pane, choose **S3 Protection**.
  2. On the **S3 Protection** page, choose **Edit**.
  3. Choose **Configure accounts manually**.
  4. Select **Automatically enable for new member accounts**. This step ensures that whenever a new account joins your organization, S3 Protection will be automatically enabled for their account. Only the organization delegated GuardDuty administrator account can modify this configuration.
  5. Choose **Save**.

- Using the **Accounts** page:

  1. In the navigation pane, choose **Accounts**.
  2. On the **Accounts** page, choose **Auto-enable** preferences.
  3. In the **Manage auto-enable preferences** window, select **Enable for new accounts** under **S3 Protection**.
  4. Choose **Save**.

API/CLI

- To selectively enable S3 Protection for your member accounts, invoke the UpdateOrganizationConfiguration API operation using your own *detector ID*.
- The following example shows how you can enable S3 Protection for a single member account. Set the preferences to automatically enable or disable the protection plan in that Region for new accounts (`NEW`) that join the organization, all the accounts (`ALL`), or none of the accounts (`NONE`) in the organization. For more information, see autoEnableOrganizationMembers. Based on your preference, you may need to replace `NEW` with `ALL` or `NONE`.

  To find the `detectorId` for your account and current Region, see the **Settings** page in the https://console.aws.amazon.com/guardduty/ console, or run the ListDetectors API.

```
aws guardduty update-organization-configuration --detector-
id 12abc34d567e8fa901bc2d34e56789f0 --auto-enable --features '[{"Name":
 "S3_DATA_EVENTS", "autoEnable": "NEW"}]'
```

- When the code has successfully executed, it returns an empty list of `UnprocessedAccounts`. If there were any problems changing the detector settings for an account, that account ID is listed along with a summary of the issue.

Selectively enable S3 Protection in member accounts

Choose your preferred access method to selectively enable S3 Protection for member accounts.

Console

1. Open the GuardDuty console at https://console.aws.amazon.com/guardduty/.

   Make sure to use the delegated GuardDuty administrator account credentials.
2. In the navigation pane, choose **Accounts**.

   On the **Accounts** page, review the **S3 Protection** column for the status of your member account.
3. **To selectively enable S3 Protection**

   Select the account for which you want to enable S3 Protection. You can select multiple accounts at a time. In the **Edit Protection Plans** dropdown menu, choose **S3Pro**, and then choose the appropriate option.

API/CLI

To selectively enable S3 Protection for your member accounts, run the updateMemberDetectors API operation using your own detector ID. The following example shows how you can enable S3 Protection for a single member account. To disable it, replace `true` with `false`.

To find the `detectorId` for your account and current Region, see the **Settings** page in the https://console.aws.amazon.com/guardduty/ console, or run the ListDetectors API.

```
aws guardduty update-member-detectors --detector-
id 12abc34d567e8fa901bc2d34e56789f0 --account-ids 123456789012 --features
 '[{"Name" : "S3_DATA_EVENTS", "Status" : "ENABLED"}]'
```

> **Note**
>
> You can also pass a list of account IDs separated by a space.

When the code has successfully executed, it returns an empty list of `UnprocessedAccounts`. If there were any problems changing the detector settings for an account, that account ID is listed along with a summary of the issue.

> **Note**
>
> If you use scripts to on-board new accounts and want to disable S3 Protection in your new accounts, you can modify the createDetector API operation with the optional `dataSources` object as described in this topic.

Enabling S3 Protection for a standalone account

A standalone account owns the decision to enable or disable a protection plan in their AWS account in a specific AWS Region.

If your account is associated with a GuardDuty administrator account through AWS Organizations, or by the method of invitation, this section doesn't apply to your account. For more information, see Enabling S3 Protection in multiple-account environments.

After you enable S3 Protection, GuardDuty will start monitoring AWS CloudTrail data events for the S3 buckets in your account.

Choose your preferred access method to configure S3 Protection for a standalone account.

Console

1. Sign in to the AWS Management Console and open the GuardDuty console at https://console.aws.amazon.com/guardduty/.
2. From the **Region** selector in the upper-right corner, select a Region where you want to enable S3 Protection.
3. In the navigation pane, choose **S3 Protection**.

4. The **S3 Protection** page provides the current status of S3 Protection for your account. Choose **Enable** or **Disable** to enable or disable S3 Protection at any point in time.
5. Choose **Confirm** to confirm your selection.

API/CLI

Run updateDetector by using your valid detector ID for the current Region and passing the `features` object name as S3_DATA_EVENTS set to ENABLED to enable S3 Protection, respectively.

> **Note**
>
> To find the `detectorId` for your account and current Region, see the **Settings** page in the https://console.aws.amazon.com/guardduty/ console, or run the ListDetectors API.

Alternatively, you can use AWS Command Line Interface. To enable S3 Protection, run the following command, and replace *12abc34d567e8fa901bc2d34e56789f0* with your account's detector ID and *us-east-1* with the Region where you want to enable S3 Protection.

```
aws guardduty update-detector --detector-id 12abc34d567e8fa901bc2d34e56789f0 --
region us-east-1 --features '[{"Name" : "S3_DATA_EVENTS", "Status" : "ENABLED"}]'
```

GuardDuty Runtime Monitoring

Runtime Monitoring observes and analyzes operating system-level, networking, and file events to help you detect potential threats in specific AWS workloads in your environment.

Supported AWS resources in Runtime Monitoring – GuardDuty had initially released Runtime Monitoring to support only Amazon Elastic Kubernetes Service (Amazon EKS) resources. Now, you can use the Runtime Monitoring feature to provide threat detection for your AWS Fargate Amazon Elastic Container Service (Amazon ECS) and Amazon Elastic Compute Cloud (Amazon EC2) resources as well.

GuardDuty doesn't support Amazon EKS clusters running on AWS Fargate.

In this document and other sections related to Runtime Monitoring, GuardDuty uses the terminology of **resource type** to refer to Amazon EKS, Fargate Amazon ECS, and Amazon EC2 resources.

Runtime Monitoring uses a GuardDuty security agent that adds visibility into runtime behavior, such as file access, process execution, command line arguments, and network connections. For each resource type that you want to monitor for potential threats, you can manage the security agent for that specific resource type either automatically or manually (with an exception to Fargate (Amazon ECS only)). Managing the security agent automatically means that you permit GuardDuty to install and update the security agent on your behalf. On the other hand, when you manage the security agent for your resources manually, you are responsible for installing and updating the security agent, as needed.

With this extended capability, GuardDuty can help you identify and respond to potential threats that may target applications and data running in your individual workloads and instances. For example, a threat can potentially start by compromising a single container that runs a vulnerable web application. This web application might have access permissions to the underlying containers and workloads. In this scenario, incorrectly configured credentials could potentially lead to a broader access to the account, and the data stored within it.

By analyzing the runtime events of the individual containers and workloads, GuardDuty can potentially identify compromise of a container and associated AWS credentials in an initial phase, and detect attempts to escalate privileges, suspicious API requests, and malicious access to the data in your environment.

Contents

- How it works
- How does 30-day free trial work in Runtime Monitoring
- Prerequisites to enabling Runtime Monitoring
- Enabling GuardDuty Runtime Monitoring
- Managing GuardDuty security agents
- Reviewing runtime coverage statistics and troubleshooting issues
- Setting up CPU and memory monitoring
- Using shared VPC with Runtime Monitoring
- Using Infrastructure as Code (IaC) with GuardDuty automated security agents
- Collected runtime event types that GuardDuty uses
- Amazon ECR repository hosting GuardDuty agent
- Two security agents on same underlying host
- EKS Runtime Monitoring in GuardDuty
- GuardDuty security agent release versions
- Disabling, uninstalling, and cleaning up resources in Runtime Monitoring

How it works

To use Runtime Monitoring, you must enable Runtime Monitoring and then manage the GuardDuty security agent. The following list explains this two-step process:

1. **Enable Runtime Monitoring** for your account so that GuardDuty can accept the runtime events that it receives from your Amazon EC2 instances, Amazon ECS clusters, and Amazon EKS workloads.
2. **Manage GuardDuty agent** for the individual resources for which you want to monitor the runtime behavior. Based on the resource type, you can choose to:
   - Use automated agent configuration, where GuardDuty manages the agent deployment and automatically an Amazon Virtual Private Cloud (Amazon VPC) endpoint.
   - Install agent manually, which requires you to create the VPC endpoint as a prerequisite.

   The security agent uses VPC endpoint to deliver events to GuardDuty, ensuring that the data remains within the AWS network. This approach enhances security and allows GuardDuty to monitor and analyze runtime behavior across your resources (Amazon EKS, Amazon EC2, and AWS Fargate-Amazon ECS). GuardDuty uses Instance identity roles that authenticates the security agent for each resource type to send the associated runtime events to the VPC endpoint.

> **Note**
>
> GuardDuty doesn't make the runtime events accessible to you.

When you manage the security agent (either manually or through GuardDuty) in EKS Runtime Monitoring or Runtime Monitoring for EC2 instances, and GuardDuty is presently deployed on an Amazon EC2 instance and receives the Collected runtime event types from this instance, GuardDuty will not charge your AWS account for the analysis of VPC flow logs from this Amazon EC2 instance. This helps GuardDuty avoid double usage cost in the account.

The following topics explain how enabling Runtime Monitoring and managing GuardDuty security agent works differently for each resource type.

Contents

- How Runtime Monitoring works with Amazon EKS clusters
- How Runtime Monitoring works with Amazon EC2 instances
- How Runtime Monitoring works with Fargate (Amazon ECS only)
- After you enable Runtime Monitoring

How Runtime Monitoring works with Amazon EKS clusters

Runtime Monitoring uses an EKS add-on `aws-guardduty-agent`, also called as GuardDuty security agent. After GuardDuty security agent gets deployed on your EKS clusters, GuardDuty is able to receive runtime events for these EKS clusters.

> **Notes**
>
> Runtime Monitoring **supports** Amazon EKS clusters running on Amazon EC2 instances and Amazon EKS Auto Mode.
> Runtime Monitoring **doesn't support** Amazon EKS clusters with Amazon EKS Hybrid Nodes, and those running on AWS Fargate.

For information about these Amazon EKS features, see What is Amazon EKS? in the Amazon EKS User Guide.

You can monitor the runtime events of your Amazon EKS clusters at either account or cluster level. You can manage the GuardDuty security agent for only those Amazon EKS clusters that you want to monitor for threat detection. You can manage the GuardDuty security agent either manually or by allowing GuardDuty to manage it on your behalf, by using Automated agent configuration.

When you use the automated agent configuration approach to allow GuardDuty to manage the deployment of the security agent on your behalf, it will automatically **create an Amazon Virtual Private Cloud (Amazon VPC) endpoint**. The security agent delivers the runtime events to GuardDuty by using this Amazon VPC endpoint.

Along with the VPC endpoint, GuardDuty also creates a new security group. The inbound (ingress) rules control the traffic that's allowed to reach the resources, that are associated with the security group. GuardDuty adds inbound rules that match the VPC CIDR range for your resource, and also adapts to it when the CIDR range changes. For more information, see VPC CIDR range in the *Amazon VPC User Guide*.

Notes

- There is no additional cost for the usage of the VPC endpoint.
- Working with centralized VPC with automated agent – When you use GuardDuty automated agent configuration for a resource type, GuardDuty will create a VPC endpoint on your behalf for all the VPCs. This includes the centralized VPC and spoke VPCs. GuardDuty doesn't support creating a VPC endpoint only for the centralized VPC. For more information about how the centralized VPC works, see Interface VPC endpoints in the *AWS Whitepaper - Building a Scalable and Secure Multi-VPC AWS Network Infrastructure*.

Approaches to manage GuardDuty security agent in Amazon EKS clusters

Prior to September 13, 2023, you could configure GuardDuty to manage the security agent at the account level. This behavior indicated that by default, GuardDuty will manage the security agent on all the EKS clusters that belong to an AWS account. Now, GuardDuty provides a granular capability to help you choose the EKS clusters where you want GuardDuty to manage the security agent.

When you choose to Manage GuardDuty security agent manually, you can still select the EKS clusters that you want to monitor. However, to manage the agent manually, creating a Amazon VPC endpoint for your AWS account is a prerequisite.

> **Note**
>
> Regardless of the approach that you use to manage the GuardDuty security agent, EKS Runtime Monitoring is always enabled at the account level.

Topics

- Manage security agent through GuardDuty
- Manage GuardDuty security agent manually

Manage security agent through GuardDuty

GuardDuty deploys and manages the security agent on your behalf. At any point in time, you can monitor the EKS clusters in your account by using one of the following approaches.

Topics

- Monitor all EKS clusters
- Exclude selective EKS clusters
- Include selective EKS clusters

Monitor all EKS clusters

Use this approach when you want GuardDuty to deploy and manage the security agent for all the EKS clusters in your account. By default, GuardDuty will also deploy the security agent on a potentially new EKS cluster created in your account.

Impact of using this approach

- GuardDuty creates an Amazon Virtual Private Cloud (Amazon VPC) endpoint through which the GuardDuty security agent delivers the runtime events to GuardDuty. There is no additional cost for the creation of the Amazon VPC endpoint when you manage the security agent through GuardDuty.

- It is required that your worker node has a valid network path to an active `guardduty-data` VPC endpoint. GuardDuty deploys the security agent on your EKS clusters. Amazon Elastic Kubernetes Service (Amazon EKS) will coordinate the deployment of the security agent on the nodes within the EKS clusters.
- On the basis of IP availability, GuardDuty selects the subnet to create a VPC endpoint. If you use advanced network topologies, you must validate that the connectivity is possible.

Exclude selective EKS clusters

Use this approach when you want GuardDuty to manage the security agent for all EKS clusters in your account but exclude selective EKS clusters. This method uses a tag-based[1] approach wherein you can tag the EKS clusters for which you don't want to receive the runtime events. The pre-defined tag must have `GuardDutyManaged`-`false` as the key-value pair.

Impact of using this approach

This approach requires that you to enable GuardDuty agent auto-management only after adding tags to the EKS clusters that you want to exclude from monitoring.

Therefore, the impact when you Manage security agent through GuardDuty applies to this approach too. When you add tags prior to enabling GuardDuty agent auto-management, GuardDuty will neither deploy nor manage the security agent for the EKS clusters that are excluded from monitoring.

Considerations

- You must add the tag key-value pair as `GuardDutyManaged`:`false` for the selective EKS clusters before enabling Automated agent configuration otherwise, the GuardDuty security agent will be deployed on all the EKS clusters until you use the tag.
- You must prevent the tags from being modified, except by trusted identities.

> **Important**
>
> Manage permissions for modifying the value of the `GuardDutyManaged` tag for your EKS cluster by using service control policies or IAM policies. For more information, see

> Service control policies (SCPs) in the *AWS Organizations User Guide* or Control access to AWS resources in the *IAM User Guide*.

- For a potentially new EKS cluster that you don't want to monitor, make sure to add the `GuardDutyManaged-false` key-value pair at the time of creating this EKS cluster.
- This approach will also have the same consideration as specified for Monitor all EKS clusters.

Include selective EKS clusters

Use this approach when you want GuardDuty to deploy and manage the updates to the security agent only for selective EKS clusters in your account. This method uses a tag-based[1] approach wherein you can tag the EKS cluster for which you want to receive the runtime events.

Impact of using this approach

- By using inclusion tags, GuardDuty will automatically deploy and manage the security agent only for the selective EKS clusters that are tagged with `GuardDutyManaged-true` as the key-value pair.
- Using this approach will also have the same impact as specified for Monitor all EKS clusters.

Considerations

- If the value of the `GuardDutyManaged` tag is not set to `true`, the inclusion tag will not work as expected and this may impact monitoring your EKS cluster.
- To ensure that your selective EKS clusters are being monitored, you need to prevent the tags from being modified, except by trusted identities.

> **Important**
>
> Manage permissions for modifying the value of the `GuardDutyManaged` tag for your EKS cluster by using service control policies or IAM policies. For more information, see Service control policies (SCPs) in the *AWS Organizations User Guide* or Control access to AWS resources in the *IAM User Guide*.

- For a potentially new EKS cluster that you don't want to monitor, make sure to add the `GuardDutyManaged-false` key-value pair at the time of creating this EKS cluster.
- This approach will also have the same consideration as specified for Monitor all EKS clusters.

[1]For more information about tagging selective EKS clusters, see Tagging your Amazon EKS resources in the Amazon EKS User Guide.

Manage GuardDuty security agent manually

Use this approach when you want deploy and manage the GuardDuty security agent on all of your EKS clusters manually. Ensure that EKS Runtime Monitoring is enabled for your accounts. The GuardDuty security agent may not work as expected if you don't enable EKS Runtime Monitoring.

Impact of using this approach

You will need to coordinate the deployment of the GuardDuty security agent within your EKS clusters across all accounts and AWS Regions where this feature is available. You will also need to update the agent version when GuardDuty releases it. For more information about agent versions for EKS, see GuardDuty security agent versions for Amazon EKS resources.

Considerations

You must support secure data flow while monitoring for and addressing coverage gaps as new clusters and workloads are continuously deployed.

How Runtime Monitoring works with Amazon EC2 instances

Your Amazon EC2 instances can run multiple types of applications and workloads in your AWS environment. When you enable Runtime Monitoring and manage the GuardDuty security agent, GuardDuty helps you detect threats in your existing Amazon EC2 instances and potentially new ones. This feature also supports Amazon ECS managed Amazon EC2 instances.

Enabling Runtime Monitoring makes GuardDuty ready to consume runtime events from currently running and new processes within Amazon EC2 instances. GuardDuty requires a security agent to send runtime events from your EC2 instance to GuardDuty.

For Amazon EC2 instances, GuardDuty security agent operates at the instance level. You can decide if you want to monitor all or selective Amazon EC2 instances in your account. If you want to manage selective instances, the security agent is required only for these instances.

GuardDuty can also consume runtime events from new tasks and existing tasks running in Amazon EC2 instances within Amazon ECS clusters.

To install the GuardDuty security agent, Runtime Monitoring provides the following two options:

- [Use automated agent configuration (recommended)](), or
- [Manage security agent manually]()

Use automated agent configuration through GuardDuty (recommended)

Use automated agent configuration that permits GuardDuty to install the security agent on your Amazon EC2 instances on your behalf. GuardDuty also manages the updates to the security agent.

By default, GuardDuty installs the security agent on all the instances in your account. If you'd want GuardDuty to install and manage the security agent for selected EC2 instances only, add inclusion or exclusion tags to your EC2 instances, as needed.

Sometimes, you may not want to monitor runtime events for all the Amazon EC2 instances that belong to your account. For cases when you want to monitor the runtime events for a limited number of instances, add an inclusion tag as `GuardDutyManaged:true` to these selected instances. Starting with the availability of automated agent configuration for Amazon EC2, if your EC2 instance has an inclusion tag (`GuardDutyManaged:true`), GuardDuty will honor the tag and manage the security agent for the selected instances even when you do not explicitly enable automated agent configuration.

On the other hand, if there are a limited number of EC2 instances for which you don't want to monitor runtime events, add an exclusion tag (`GuardDutyManaged:false`) to these selected instances. GuardDuty will honor the exclusion tag by **neither** installing **nor** managing the security agent for these EC2 resources.

Impact

When you use automated agent configuration in an AWS account or an organization, you permit GuardDuty to take the following steps on your behalf:

- GuardDuty creates one SSM association for all your Amazon EC2 instances that are SSM managed and appear under **Fleet Manager** in the https://console.aws.amazon.com/systems-manager/ console.
- Using inclusion tags with automated agent configuration disabled – After enabling Runtime Monitoring, when you don't enable automated agent configuration but add inclusion tag to your Amazon EC2 instance, it means that you are permitting GuardDuty to manage the security agent on your behalf. SSM association will then install the security agent in each instance that has the inclusion tag (`GuardDutyManaged:true`).

- If you enable automated agent configuration – The SSM association will then install the security agent in all the EC2 instances that belong to your account.
- Using exclusion tags with automated agent configuration – Before you enable automated agent configuration, when you add exclusion tag to your Amazon EC2 instance, it means that you are permitting GuardDuty to prevent installing and managing the security agent for this selected instance.

  Now, when you enable automated agent configuration, the SSM association will install and manage the security agent in all the EC2 instances except the ones that are tagged with the exclusion tag.
- GuardDuty creates VPC endpoints in all the VPCs, including shared VPCs, as long as there is at least one Linux EC2 instance in that VPC that are not in the terminated or shutting-down instance states. This includes the centralized VPC and spoke VPCs. GuardDuty doesn't support creating a VPC endpoint only for the centralized VPC. For more information about how the centralized VPC works, see Interface VPC endpoints in the *AWS Whitepaper - Building a Scalable and Secure Multi-VPC AWS Network Infrastructure*.

  For information about different instance states, see Instance lifecycle in the *Amazon EC2 User Guide*.

  GuardDuty also supports Using shared VPC with Runtime Monitoring. When all the prerequisites are considered for your organization and AWS account, GuardDuty will use the shared VPC to receive runtime events.

  > **Note**
  >
  > There is no additional cost for the usage of the VPC endpoint.
- Along with the VPC endpoint, GuardDuty also creates a new security group. The inbound (ingress) rules control the traffic that's allowed to reach the resources, that are associated with the security group. GuardDuty adds inbound rules that match the VPC CIDR range for your resource, and also adapts to it when the CIDR range changes. For more information, see VPC CIDR range in the *Amazon VPC User Guide*.

Manage security agent manually

There are two ways to manage the security agent for Amazon EC2 manually:

- Use GuardDuty managed documents in AWS Systems Manager to install the security agent on your Amazon EC2 instances that are already SSM managed.

  Whenever you launch a new Amazon EC2 instance, ensure that it is SSM enabled.
- Use RPM package manager (RPM) scripts to install the security agent on your Amazon EC2 instances, whether or not they are SSM managed.

Next step

To get started with Runtime Monitoring configuration to monitor your Amazon EC2 instances, see Prerequisites for Amazon EC2 instance support.

How Runtime Monitoring works with Fargate (Amazon ECS only)

When you enable Runtime Monitoring, GuardDuty becomes ready to consume the runtime events from a task. These tasks run within the Amazon ECS clusters, which in turn run on the AWS Fargate instances. For GuardDuty to receive these runtime events, you must use the fully-managed, dedicated security agent.

You can allow GuardDuty to manage the GuardDuty security agent on your behalf, by using Automated agent configuration for an AWS account or an organization. GuardDuty will start deploying the security agent to the new Fargate tasks that are launched in your Amazon ECS clusters. The following list specifies what to expect when you enable the GuardDuty security agent.

Impact of enabling GuardDuty security agent

GuardDuty creates a virtual private cloud (VPC) endpoint and security group

- When you deploy the GuardDuty security agent, GuardDuty will create a VPC endpoint through which the security agent delivers the runtime events to GuardDuty.

  Along with the VPC endpoint, GuardDuty also creates a new security group. The inbound (ingress) rules control the traffic that's allowed to reach the resources, that are associated with the security group. GuardDuty adds inbound rules that match the VPC CIDR range for your resource, and also adapts to it when the CIDR range changes. For more information, see VPC CIDR range in the *Amazon VPC User Guide*.
- Working with centralized VPC with automated agent – When you use GuardDuty automated agent configuration for a resource type, GuardDuty will create a VPC endpoint on your behalf for all the VPCs. This includes the centralized VPC and spoke VPCs. GuardDuty doesn't

support creating a VPC endpoint only for the centralized VPC. For more information about how the centralized VPC works, see Interface VPC endpoints in the *AWS Whitepaper - Building a Scalable and Secure Multi-VPC AWS Network Infrastructure*.

- There is no additional cost for the usage of the VPC endpoint.

GuardDuty adds a sidecar container

For a new Fargate task or service that starts running, a GuardDuty container (sidecar) attaches itself to each container within the Amazon ECS Fargate task. The GuardDuty security agent runs within the attached GuardDuty container. This helps GuardDuty to collect the runtime events of each container running within these tasks.

The GuardDuty sidecar container image is stored in Amazon Elastic Container Registry (Amazon ECR), with its image layers stored in Amazon S3. When your task starts, it needs to pull this image from ECR. Depending on your network configuration, this may require specific settings to ensure access to both ECR and S3. For example, if you're using security groups with restricted access, you'll need to allow access to the S3 managed prefix list. For more information on how to do this, see Prerequisites for container image access.

When you start a Fargate task, should the GuardDuty container (sidecar) be unable to launch in a healthy state, Runtime Monitoring is designed to not prevent the tasks from running.

By default, a Fargate task is immutable. GuardDuty will not deploy the sidecar when a task is already in a running state. If you want to monitor a container in an already running task, you can stop the task and start it again.

Approaches to manage GuardDuty security agent in Amazon ECS-Fargate resources

Runtime Monitoring provides you the option to detect potential security threats on either all of the Amazon ECS clusters (account level) or selective clusters (cluster level) in your account. When you enable Automated agent configuration for each Amazon ECS Fargate task that will run, GuardDuty will add a sidecar container for each container workload within that task. The GuardDuty security agent gets deployed to this sidecar container. This is how GuardDuty gets visibility into the runtime behavior of the containers inside the Amazon ECS tasks.

Runtime Monitoring supports managing the security agent for your Amazon ECS clusters (AWS Fargate) only through GuardDuty. There is no support for managing the security agent manually on Amazon ECS clusters.

Before you configure your accounts, assess if you want to monitor the runtime behavior of all the containers that belong to the Amazon ECS tasks, or include or exclude specific resources. Consider the following approaches.

Monitor for all Amazon ECS clusters

This approach will help you detect potential security threats at account level. Use this approach when you want GuardDuty to detect potential security threats for all the Amazon ECS clusters that belong to your account.

Exclude specific Amazon ECS clusters

Use this approach when you want GuardDuty to detect potential security threats for most of the Amazon ECS clusters in your AWS environment but exclude some of the clusters. This approach helps you monitor the runtime behavior of the containers within your Amazon ECS tasks at the cluster level. For example, the number of Amazon ECS clusters that belong to your account are 1000. However, you want to monitor only 930 Amazon ECS clusters.

This approach requires you to add a pre-defined GuardDuty tag to the Amazon ECS clusters that you don't want to monitor. For more information, see Managing automated security agent for Fargate (Amazon ECS only).

Include specific Amazon ECS clusters

Use this approach when you want GuardDuty to detect potential security threats for some of the Amazon ECS clusters. This approach helps you monitor the runtime behavior of the containers within your Amazon ECS tasks at the cluster level. For example, the number of Amazon ECS clusters that belong to your account are 1000. However, you want to monitor 230 clusters only.

This approach requires you to add a pre-defined GuardDuty tag to the Amazon ECS clusters that you want to monitor. For more information, see Managing automated security agent for Fargate (Amazon ECS only).

After you enable Runtime Monitoring

After you enable Runtime Monitoring and install GuardDuty security agent in your standalone account or multiple member accounts, you can take the following steps to ensure that the protection plan setting is working as expected, and monitor how much memory and CPU does GuardDuty security agent uses.

Assess runtime coverage

GuardDuty recommends you to continuously assess the coverage status of the resource where you have deployed the security agent. The coverage status could be either **Healthy** or **Unhealthy**. A **Healthy** coverage status indicates that GuardDuty is receiving the runtime events from the corresponding resource when there is an operating system-level activity.

When the coverage status becomes **Healthy** for the resource, GuardDuty is able to receive the runtime events and analyze them for threat detection. When GuardDuty detects a potential security threat in the tasks or applications running in your container workloads and instances, GuardDuty generates GuardDuty Runtime Monitoring finding types.

You can also configure an Amazon EventBridge (EventBridge) to receive a notification when the coverage status changes from **Unhealthy** to **Healthy** and otherwise. For more information, see Reviewing runtime coverage statistics and troubleshooting issues.

Set up CPU and memory monitoring for GuardDuty security agent

After you have assessed that the coverage status shows as **Healthy**, you can evaluate the performance of the security agent for your resource type. For Amazon EKS clusters that have the security agent release v1.5 or above, GuardDuty supports configuring the parameters of the (add-on) security agent. For more information, see Setting up CPU and memory monitoring.

GuardDuty detects potential threats

As GuardDuty starts to receive the runtime events for your resource, it starts analyzing those events. When GuardDuty detects a potential security threat in any of your Amazon EC2 instances, Amazon ECS clusters, or Amazon EKS clusters, it generates one or more GuardDuty Runtime Monitoring finding types. You can access the finding details to view the impacted resource details.

How does 30-day free trial work in Runtime Monitoring

The 30-day free trial period works differently for the new GuardDuty accounts and the existing accounts that have already enabled EKS Runtime Monitoring prior to when Runtime Monitoring capability extended to Amazon EC2 instances and AWS Fargate (Amazon ECS only).

I am using GuardDuty trial period or I have never enabled EKS Runtime Monitoring

The following list explains how the 30-day free trial period works if you're either using the GuardDuty 30-day trial period or have never enabled EKS Runtime Monitoring:

- When you enable GuardDuty for the first time, Runtime Monitoring and EKS Runtime Monitoring will not be enabled by default.

  When you enable Runtime Monitoring for your account or organization, make sure to also configure the GuardDuty security agent for the resource that you want to monitor for threat detection. For example, if you want to use Runtime Monitoring for your Amazon EC2 instances, then after you enable Runtime Monitoring, you must also configure the security agent for Amazon EC2. You can choose to do this either manually or automatically through GuardDuty.
- The Runtime Monitoring protection plan is enabled at the **account level**. The 30-day free trial period works at the **resource level**. After the GuardDuty security agent gets deployed to a specific resource type, the 30-day free trial starts when GuardDuty receives its **first runtime event** associated with this resource type. For example, you have deployed the GuardDuty agent at the resource level (for Amazon EC2 instance, Amazon ECS cluster, and Amazon EKS cluster). When GuardDuty receives the first runtime event for an Amazon EC2 instance, the 30-day free trial will start for Amazon EC2 only.
- When you want to enable only EKS Runtime Monitoring – When you enable GuardDuty for the first time, EKS Runtime Monitoring is not enabled by default (after the release of Runtime Monitoring). You will need to enable EKS Runtime Monitoring. To use it optimally, make sure that you either manage the GuardDuty security agent manually or enable automated agent configuration so that GuardDuty manages the agent on your behalf. Your 30-day free trial period for EKS Runtime Monitoring starts when GuardDuty receives its first runtime event for the Amazon EKS resource.

I enabled EKS Runtime Monitoring prior to the launch of Runtime Monitoring

Use this section only when EKS Runtime Monitoring was enabled for your AWS account, and now you want to migrate to Runtime Monitoring.

The following list includes scenarios that might apply to your use case of enabling Runtime Monitoring:

- For an existing GuardDuty account that has the EKS Runtime Monitoring protection plan enabled and uses the GuardDuty console experience to use this protection plan – With the announcement of Runtime Monitoring, the EKS Runtime Monitoring console experience has now been consolidated into Runtime Monitoring. Your existing configuration for EKS Runtime Monitoring remains the same. You can continue to use the API/CLI support to perform operations associated with EKS Runtime Monitoring.
- To use EKS Runtime Monitoring as a part of Runtime Monitoring, you will need to configure Runtime Monitoring for your account or organization. To keep the same configuration for Runtime Monitoring, see Migrating from EKS Runtime Monitoring to Runtime Monitoring. However, this will not impact your 30-day free trial for Amazon EKS resource.
- The Runtime Monitoring protection plan is enabled at the account level per Region. After the GuardDuty security agent gets deployed to one of the specified resource types (Amazon EC2 instance and Amazon ECS cluster), the 30-day free trial starts when GuardDuty receives the first runtime event associated with the resource. There is a 30-day free trial associated with each resource type.

  For example, after enabling Runtime Monitoring, you choose to deploy the GuardDuty agent only on Amazon EC2 instance, the 30-day free trial for this resource will start only when GuardDuty receives its first runtime event for an Amazon EC2 instance. Later, when you deploy the GuardDuty agent for Fargate (Amazon ECS only), the 30-day free trial for this resource will start only when GuardDuty receives its first runtime event for Amazon ECS cluster. Considering you already have EKS Runtime Monitoring enabled for your account, GuardDuty doesn't reset the 30-day free trial for an Amazon EKS resource.

Prerequisites to enabling Runtime Monitoring

To enable Runtime Monitoring and manage the GuardDuty security agent, you must meet the prerequisites for each resource type that you want to monitor for threat detection. Each resource type has different prerequisites. For example, GuardDuty supports different OS distributions based on the resource type.

When you want to monitor only Amazon EC2 resources, you will follow the prerequisites for Amazon EC2 instances. If at a later time, you choose to monitor Amazon EKS resources, you must follow the prerequisites specific to Amazon EKS clusters.

The following sections include prerequisites based on the resource type.

Contents

- Prerequisites for Amazon EC2 instance support
- Prerequisites for AWS Fargate (Amazon ECS only) support
- Prerequisites for Amazon EKS cluster support

Prerequisites for Amazon EC2 instance support

This section includes the prerequisites for monitoring runtime behavior of your Amazon EC2 instances. After these prerequisites are met, see Enabling GuardDuty Runtime Monitoring.

Topics

- Make EC2 instances SSM managed (for automated agent configuration only)
- Validate architectural requirements
- Validating your organization service control policy in a multi-account environment
- When using automated agent configuration
- CPU and memory limit for GuardDuty agent
- Next step

Make EC2 instances SSM managed (for automated agent configuration only)

GuardDuty uses AWS Systems Manager (SSM) to automatically deploy, install, and manage the security agent on your instances. If you plan to manually install and manage the GuardDuty agent, SSM is not required.

To manage your Amazon EC2 instances with Systems Manager, see Setting up Systems Manager for Amazon EC2 instances in the *AWS Systems Manager User Guide*.

Validate architectural requirements

The architecture of your OS distribution might impact how the GuardDuty security agent will behave. You must meet the following requirements before using Runtime Monitoring for Amazon EC2 instances:

- Kernel support includes `eBPF`, `Tracepoints` and `Kprobe`. For CPU architectures, Runtime Monitoring supports AMD64 (`x64`) and ARM64 (Graviton2 and above)[1].

  The following table shows the OS distribution that has been verified to support the GuardDuty security agent for Amazon EC2 instances.

OS distribution[2]	Kernel version[3]
Amazon Linux 2	5.4[4], 5.10[4], 5.15
Amazon Linux 2023	5.4[4], 5.10[4], 5.15, 6.1, 6.5, 6.8, 6.12
Ubuntu 20.04 and Ubuntu 22.04	5.4[4], 5.10[4], 5.15, 6.1, 6.5, 6.8
Ubuntu 24.04	6.8
Debian 11 and Debian 12	5.4[4], 5.10[4], 5.15, 6.1, 6.5, 6.8
RedHat 9.4	5.14
Fedora 34.0	5.11, 5.17
Fedora 40	6.8
Fedora 41	6.12
CentOS Stream 9	5.14
Oracle Linux 8.9	5.15
Oracle Linux 9.3	5.15
Rocky Linux 9.5	5.14

1. Runtime Monitoring for Amazon EC2 resources doesn't support the first generation Graviton instance such as A1 instance types.

2.

Support for various operating systems - GuardDuty has verified Runtime Monitoring support for the operating distribution listed in the preceding table. While the GuardDuty security agent may run on operating systems not listed in the preceding table, the GuardDuty team cannot guarantee the expected security value.

3. For any kernel version, you must set the `CONFIG_DEBUG_INFO_BTF` flag to `y` (meaning *true*). This is required so that the GuardDuty security agent can run as expected.

4. For kernel versions 5.10 and earlier, the GuardDuty security agent uses locked memory in RAM (`RLIMIT_MEMLOCK`) to function as expected. If your system's `RLIMIT_MEMLOCK` value is set too low, GuardDuty recommends setting both hard and soft limits to at least 32 MB. For information about verifying and modifying the default `RLIMIT_MEMLOCK` value, see Viewing and updating RLIMIT_MEMLOCK values.

- Additional requirements - Only if you have Amazon ECS/Amazon EC2

  For Amazon ECS/Amazon EC2, we recommend that you use the latest Amazon ECS-optimized AMIs (dated September 29, 2023 or later), or use Amazon ECS agent version v1.77.0.

Viewing and updating `RLIMIT_MEMLOCK` values

When your system's `RLIMIT_MEMLOCK` limit is set too low, GuardDuty security agent may not perform as designed. GuardDuty recommends that both hard and soft limits must be at least 32 MB. If you don't update the limits, GuardDuty will be unable to monitor the runtime events for your resource. When `RLIMIT_MEMLOCK` is above the minimum stated limits, it becomes optional for you to update these limits.

You can modify the default `RLIMIT_MEMLOCK` value either before or after installing the GuardDuty security agent.

To view `RLIMIT_MEMLOCK` values

1. Run `ps aux | grep guardduty`. This will output the process ID (`pid`).
2. Copy the process ID (`pid`) from the output of the previous command.
3. Run `grep "Max locked memory" /proc/pid/limits` after replacing the `pid` with the process ID copied from the previous step.

   This will display the maximum locked memory for running the GuardDuty security agent.

To update `RLIMIT_MEMLOCK` values

1. If the `/etc/systemd/system.conf.d/NUMBER-limits.conf` file exists, then comment out the line of `DefaultLimitMEMLOCK` from this file. This file sets a default `RLIMIT_MEMLOCK` with high priority, which overwrites your settings in the `/etc/systemd/system.conf` file.
2. Open the `/etc/systemd/system.conf` file and uncomment the line that has `#DefaultLimitMEMLOCK=`.
3. Update the default value by providing both hard and soft `RLIMIT_MEMLOCK` limits to at least 32MB. The update should look like this: `DefaultLimitMEMLOCK=32M:32M`. The format is `soft-limit:hard-limit`.
4. Run `sudo reboot`.

Validating your organization service control policy in a multi-account environment

If you have set up a service control policy (SCP) to manage permissions in your organization, validate that permissions boundary allows the `guardduty:SendSecurityTelemetry` action. It is required for GuardDuty to support Runtime Monitoring across different resource types.

If you are a member account, connect with the associated delegated administrator. For information about managing SCPs for your organization, see Service control policies (SCPs).

When using automated agent configuration

To Use automated agent configuration (recommended), your AWS account must meet the following prerequisites:

- When using inclusion tags with automated agent configuration, for GuardDuty to create an SSM association for a new instance, ensure that the new instance is SSM managed and shows up under **Fleet Manager** in the https://console.aws.amazon.com/systems-manager/ console.
- When using exclusion tags with automated agent configuration:
  - Add the `GuardDutyManaged:false` tag before configuring the GuardDuty automated agent for your account.

    Ensure that you add the exclusion tag to your Amazon EC2 instances before you launch them. Once you have enabled automated agent configuration for Amazon EC2, any EC2 instance that launches without an exclusion tag will be covered under GuardDuty automated agent configuration.

- Enable **Allow tags in metadata** setting for your instances. This setting is required because GuardDuty needs to read the exclusion tag from the instance metadata service (IMDS) to determine whether it should exclude the instance from agent installation. For more information, see Enable access to tags in instance metadata in the *Amazon EC2 User Guide*.

CPU and memory limit for GuardDuty agent

CPU limit

The maximum CPU limit for the GuardDuty security agent associated with Amazon EC2 instances is 10 percent of the total vCPU cores. For example, if your EC2 instance has 4 vCPU cores, then the security agent can use a maximum of 40 percent out of the total available 400 percent.

Memory limit

From the memory associated with your Amazon EC2 instance, there is a limited memory that the GuardDuty security agent can use.

The following table shows the memory limit.

Memory of the Amazon EC2 instance	Maximum memory for GuardDuty agent
Less than 8 GB	128 MB
Less than 32 GB	256 MB
More than or equal to 32 GB	1 GB

Next step

The next step is to configure Runtime Monitoring and also manage the security agent (automatically or manually).

Prerequisites for AWS Fargate (Amazon ECS only) support

This section includes the prerequisites for monitoring runtime behavior of your Fargate-Amazon ECS resources. After these prerequisites are met, see Enabling GuardDuty Runtime Monitoring.

Topics

- Validating architectural requirements
- Prerequisites for container image access
- Validating your organization service control policy in a multi-account environment
- Validating role permissions and policy permissions boundary
- CPU and memory limits

Validating architectural requirements

The platform that you use may impact how GuardDuty security agent supports GuardDuty in receiving the runtime events from your Amazon ECS clusters. You must validate that you're using one of the verified platforms.

Initial considerations:

> The AWS Fargate platform for your Amazon ECS clusters must be Linux. The corresponding platform version must be at least `1.4.0`, or `LATEST`. For more information about the platform versions, see Linux platform versions in the *Amazon Elastic Container Service Developer Guide*.
>
> The Windows platform versions are not yet supported.

Verified platforms

The OS distribution and CPU architecture impacts the support provided by the GuardDuty security agent. The following table shows the verified configuration for deploying the GuardDuty security agent and configuring Runtime Monitoring.

OS distribution[1]	Kernel support	CPU architecture x64 (AMD64)	CPU architecture Graviton (ARM64)
Linux	eBPF, Tracepoints, Kprobe	Supported	Supported

[1]Support for various operating systems - GuardDuty has verified Runtime Monitoring support for the operating distribution listed in the preceding table. While the GuardDuty security agent may run on operating systems not listed in the preceding table, the GuardDuty team cannot guarantee the expected security value.

Prerequisites for container image access

The following prerequisites help you access the GuardDuty sidecar container image from the Amazon ECR repository.

Permissions requirements

The task execution role requires certain Amazon Elastic Container Registry (Amazon ECR) permissions to download the GuardDuty security agent container image:

```
...
      "ecr:GetAuthorizationToken",
      "ecr:BatchCheckLayerAvailability",
      "ecr:GetDownloadUrlForLayer",
      "ecr:BatchGetImage",
...
```

To further restrict the Amazon ECR permissions, you can add the Amazon ECR repository URI that hosts the GuardDuty security agent for AWS Fargate (Amazon ECS only). For more information, see Amazon ECR repository hosting GuardDuty agent.

You can either use the AmazonECSTaskExecutionRolePolicy managed policy or add the above permissions to your `TaskExecutionRole` policy.

Task definition configuration

When creating or updating Amazon ECS services, you need to provide subnet information in your task definition:

Running the CreateService and UpdateService APIs in the *Amazon Elastic Container Service API Reference* requires you to pass the subnet information. For more information, see Amazon ECS task definitions in the *Amazon Elastic Container Service Developer Guide*.

Network connectivity requirements

You must ensure network connectivity to download the GuardDuty container image from Amazon ECR. This requirement is specific to GuardDuty because it uses Amazon ECR to host its security agent. Depending on your network configuration, you need to implement one of the following options:

Option 1 - Using public network access (if available)

> If your Fargate tasks run in subnets with outbound internet access, no additional network configuration is required.

Option 2 - Using Amazon VPC endpoints (for private subnets)

> If your Fargate tasks run in private subnets without internet access, you must configure VPC endpoints for ECR to ensure that the ECR repository URI that hosts the GuardDuty security agent is network accessible. Without these endpoints, tasks in private subnets cannot download the GuardDuty container image.
>
> For VPC endpoint setup instructions, see Create the VPC endpoints for Amazon ECR in the *Amazon Elastic Container Registry User Guide*.

For information about enabling Fargate to download the GuardDuty container, see Using Amazon ECR images with Amazon ECS in the *Amazon Elastic Container Registry User Guide*.

Security group configuration

The GuardDuty container images are in Amazon ECR and require Amazon S3 access. This requirement is specific to downloading container images from Amazon ECR. For tasks with restricted network access, you must configure your security groups to allow access to S3.

Add an outbound rule in your security group that allows traffic to the S3 managed prefix list (`pl-xxxxxxxx`) on port 443. To add an outbound rule, see Configure security group rules in the *Amazon VPC User Guide*.

To view your AWS-managed prefix lists in the console or describe them by using AWS Command Line Interface (AWS CLI), see AWS-managed prefix lists in the *Amazon VPC User Guide*.

Validating your organization service control policy in a multi-account environment

This section explains how to validate your service control policy (SCP) settings to ensure Runtime Monitoring works as expected across your organization.

If you have set up one or more service control policies to manage permissions in your organization, you must validate that it doesn't deny the `guardduty:SendSecurityTelemetry` action. For information about how SCPs work, see SCP evaluation in the *AWS Organizations User Guide*.

If you are a member account, connect with the associated delegated administrator. For information about managing SCPs for your organization, see Service control policies (SCPs) in the *AWS Organizations User Guide*.

Perform the following steps for all the SCPs that you have set up in your multi-account environment:

To validate `guardduty:SendSecurityTelemetry` is not denied in SCP

1. Sign in to the Organizations console at https://console.aws.amazon.com/organizations/. You must sign in as an IAM role, or sign in as the root user (not recommended) in the organization's management account.
2. In the left navigation pane, select **Policies**. Then, under **Supported policy types**, select **Service control policies**.
3. On the **Service control policies** page, choose the name of the policy that you want to validate.
4. On the policy's detail page, view the **Content** of this policy. Make sure that it doesn't deny the `guardduty:SendSecurityTelemetry` action.

   The following SCP policy is an example for *not denying* the `guardduty:SendSecurityTelemetry` action:

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
    "Effect": "Allow",
            "Action": [
                "guardduty:SendSecurityTelemetry"
            ],
            "Resource": "*"
        }
    ]
}
```

If your policy denies this action, you must update the policy. For more information, see [Update a service control policy (SCP)](#) in the *AWS Organizations User Guide*.

Validating role permissions and policy permissions boundary

Use the following steps to validate that the permissions boundaries associated with the role and its policy **doesn't** the restrict `guardduty:SendSecurityTelemetry` action.

To view permissions boundary for roles and its policy

1. Sign in to the AWS Management Console and open the IAM console at https://console.aws.amazon.com/iam/.
2. In the left navigation pane, under **Access management**, choose **Roles**.
3. On the **Roles** page, select the role *`TaskExecutionRole`* that you may have created.
4. On the selected role's page, under the **Permissions** tab, expand the policy name associated with this role. Then, validate that this policy doesn't restrict `guardduty:SendSecurityTelemetry`.
5. If the **Permissions boundary** is set, then expand this section. Then, expand each policy to review that it doesn't restrict the `guardduty:SendSecurityTelemetry` action. The policy should appear similar to this Example SCP policy.

As needed, perform one of the following actions:

- To modify the policy, select **Edit**. On the **Modify permissions** page for this policy, update the policy in the **Policy editor**. Make sure that the JSON schema remains valid. Then, choose **Next**. Then, you can review and save the changes.
- To change this permissions boundary and choose another boundary, choose **Change boundary**.
- To remove this permissions boundary, choose **Remove boundary**.

For information about managing policies, see Policies and permissions in AWS Identity and Access Management in the *IAM User Guide*.

CPU and memory limits

In the Fargate task definition, you must specify the CPU and memory value at the task level. The following table shows the valid combinations of task-level CPU and memory values, and the corresponding GuardDuty security agent maximum memory limit for the GuardDuty container.

CPU value	Memory value	GuardDuty agent maximum memory limit
256 (.25 vCPU)	512 MiB, 1 GB, 2GB	128 MB
512 (.5 vCPU)	1 GB, 2 GB, 3 GB, 4 GB	
1024 (1 vCPU)	2 GB, 3 GB, 4 GB	
	5 GB, 6 GB, 7 GB, 8 GB	
2048 (2 vCPU)	Between 4 GB and 16 GB in 1 GB increments	
4096 (4 vCPU)	Between 8 GB and 20 GB in 1 GB increments	
8192 (8 vCPU)	Between 16 GB and 28 GB in 4 GB increments	256 MB

CPU value	Memory value	GuardDuty agent maximum memory limit
	Between 32 GB and 60 GB in 4 GB increments	512 MB
16384 (16 vCPU)	Between 32 GB and 120 GB in 8 GB increments	1 GB

After you enable Runtime Monitoring and assess that the coverage status of your cluster is **Healthy**, you can set up and view the Container insight metrics. For more information, Setting up monitoring on Amazon ECS cluster.

The next step is to configure Runtime Monitoring and also configure the security agent.

Prerequisites for Amazon EKS cluster support

This section includes the prerequisites for monitoring runtime behavior of your Amazon EKS resources. These prerequisites are crucial for the GuardDuty agent to function as expected. After these prerequisites are met, see Enabling GuardDuty Runtime Monitoring to start monitoring your resources.

Support for Amazon EKS features

Runtime Monitoring **supports** Amazon EKS clusters running on Amazon EC2 instances and Amazon EKS Auto Mode.

Runtime Monitoring **doesn't support** Amazon EKS clusters with Amazon EKS Hybrid Nodes, and those running on AWS Fargate.

For information about these Amazon EKS features, see What is Amazon EKS? in the Amazon EKS User Guide.

Validating architectural requirements

The platform that you use may impact how GuardDuty security agent supports GuardDuty in receiving the runtime events from your EKS clusters. You must validate that you're using one of the verified platforms. If you're managing the GuardDuty agent manually, ensure that the Kubernetes version supports the GuardDuty agent version that is currently in use.

Verified platforms

The OS distribution, kernel version, and CPU architecture affect the support provided by the GuardDuty security agent. Kernel support includes `eBPF`, `Tracepoints` and `Kprobe`. For CPU architectures, Runtime Monitoring supports AMD64 (`x64`) and ARM64(Graviton2 and above)[1].

The following table shows the verified configuration for deploying the GuardDuty security agent and configuring EKS Runtime Monitoring.

OS distribution[2]	Kernel version[3]	Supported Kubernetes version
Bottlerocket	5.4, 5.10, 5.15, 6.1[4]	v1.23 - v1.33
Ubuntu	5.4, 5.10, 5.15, 6.1[4]	v1.21 - v1.33
Amazon Linux 2	5.4, 5.10, 5.15, 6.1[4]	v1.21 - v1.33
Amazon Linux 2023[5]	5.4, 5.10, 5.15, 6.1[4]	v1.21 - v1.33
RedHat 9.4	5.14[4]	v1.21 - v1.33
Fedora 34	5.11, 5,17	v1.21 - v1.33
Fedora 40	6.8	v1.28 - v1.33
Fedora 41	6.12	v1.28 - v1.33
CentOS Stream 9	5.14	v1.21 - v1.33

1. Runtime Monitoring for Amazon EKS clusters doesn't support the first generation Graviton instance such as A1 instance types.

2. Support for various operating systems - GuardDuty has verified Runtime Monitoring support for the operating distribution listed in the preceding table. While the GuardDuty security agent may run on operating systems not listed in the preceding table, the GuardDuty team cannot guarantee the expected security value.

3. For any kernel version, you must set the `CONFIG_DEBUG_INFO_BTF` flag to `y` (meaning *true*). This is required so that the GuardDuty security agent can run as expected.

4. Presently, with Kernel version `6.1`, GuardDuty can't generate GuardDuty Runtime Monitoring finding types that are related to Domain Name System (DNS) events.

5. Runtime Monitoring supports AL2023 with the release of the GuardDuty security agent v1.6.0 and above. For more information, see GuardDuty security agent versions for Amazon EKS resources.

Kubernetes versions supported by GuardDuty security agent

The following table shows the Kubernetes versions for your EKS clusters that are supported by GuardDuty security agent.

Amazon EKS add-on GuardDuty security agent version	Kubernetes version
v1.11.0 (latest - v1.11.0-eksbuild.2)	1.28 - 1.33
v1.10.0 (latest - v1.10.0-eksbuild.2)	1.21 - 1.33
v1.9.0 (latest - v1.9.0-eksbuild.2) v1.8.1 (latest - v1.8.1-eksbuild.2)	1.21 - 1.32
v1.7.1 v1.7.0 v1.6.1	1.21 - 1.31
v1.6.0 v1.5.0 v1.4.1 v1.4.0	1.21 - 1.29

Amazon EKS add-on GuardDuty security agent version	Kubernetes version
v1.3.1	
v1.3.0 v1.2.0	1.21 - 1.28
v1.1.0	1.21 - 1.26
v1.0.0	1.21 - 1.25

Some of the GuardDuty security agent versions will reach end of standard support.

For information about the agent release versions, see GuardDuty security agent versions for Amazon EKS resources.

CPU and memory limits

The following table shows the CPU and memory limits for the Amazon EKS add-on for GuardDuty (`aws-guardduty-agent`).

Parameter	Minimum limit	Maximum limit
CPU	200m	1000m
Memory	256 Mi	1024 Mi

When you use Amazon EKS add-on version 1.5.0 or above, GuardDuty provides the capability to configure the add-on schema for your CPU and memory values. For information about the configurable range, see Configurable parameters and values.

After you enable EKS Runtime Monitoring and assess the coverage status of your EKS clusters, you can set up and view the container insight metrics. For more information, see Setting up CPU and memory monitoring.

Validating your organization service control policy

If you have set up a service control policy (SCP) to manage permissions in your organization, validate that permissions boundary is not restricting `guardduty:SendSecurityTelemetry`. It is required for GuardDuty to support Runtime Monitoring across different resource types.

If you are a member account, connect with the associated delegated administrator. For information about managing SCPs for your organization, see Service control policies (SCPs).

Enabling GuardDuty Runtime Monitoring

Before enabling Runtime Monitoring in your account, make sure that the resource type for which you want to monitor the runtime events, supports the platforms requirements. For more information, see Prerequisites.

If you have been using EKS Runtime Monitoring prior to the launch of Runtime Monitoring, you can use the APIs to check and update the existing configuration for EKS Runtime Monitoring. You can also migrate your existing configuration from EKS Runtime Monitoring to Runtime Monitoring. For more information, see Migrating from EKS Runtime Monitoring to Runtime Monitoring.

> **Note**
>
> Presently, this documentation provides steps to enable Runtime Monitoring for your accounts and organization by console only. You can also enable Runtime Monitoring by using API Actions or AWS CLI for GuardDuty.

You can configure Runtime Monitoring by using the steps in the following topics.

Contents

- Enabling Runtime Monitoring for multiple-account environments
- Enabling Runtime Monitoring for a standalone account

Enabling Runtime Monitoring for multiple-account environments

In a multiple-account environments, only the delegated GuardDuty administrator account can enable or disable Runtime Monitoring for the member accounts, and manage automated agent configuration for the resource types belonging to the member accounts in their organization. The GuardDuty member accounts can't modify this configuration from their accounts. The delegated GuardDuty administrator account account manages their member accounts using AWS Organizations. For more information about multi-account environments, see Managing multiple accounts.

For delegated GuardDuty administrator account

To enable Runtime Monitoring for delegated GuardDuty administrator account

1. Sign in to the AWS Management Console and open the GuardDuty console at https://console.aws.amazon.com/guardduty/.
2. In the navigation pane, choose **Runtime Monitoring**.
3. Under the **Configuration** tab, choose **Edit** in the **Runtime Monitoring configuration** section.
4. **Using Enable for all accounts**

   If you want to enable Runtime Monitoring for all the accounts that belong to the organization, including the delegated GuardDuty administrator account, then choose **Enable for all accounts**.
5. **Using Configure accounts manually**

   If you want to enable Runtime Monitoring for each member account individually, then choose **Configure accounts manually**.

   - Choose **Enable** under the **Delegated Administrator (this account)** section.
6. For GuardDuty to receive the runtime events from one or more resource types – an Amazon EC2 instance, Amazon ECS cluster, or an Amazon EKS cluster, use the following options to manage the security agent for these resources:

   To enable GuardDuty security agent

   - Enabling automated security agent for Amazon EC2 instance
   - Managing security agent manually for Amazon EC2 resource
   - Managing automated security agent for Fargate (Amazon ECS only)
   - Managing security agent automatically for Amazon EKS resources
   - Managing security agent manually for Amazon EKS cluster

For all member accounts

To enable Runtime Monitoring for all member accounts in the organization

1. Sign in to the AWS Management Console and open the GuardDuty console at https://console.aws.amazon.com/guardduty/.

   Sign in using the delegated GuardDuty administrator account.
2. In the navigation pane, choose **Runtime Monitoring**.
3. On the Runtime Monitoring page, under the **Configuration** tab, choose **Edit** in the **Runtime Monitoring configuration** section.
4. Choose **Enable for all accounts**.
5. For GuardDuty to receive the runtime events from one or more resource types – an Amazon EC2 instance, Amazon ECS cluster, or an Amazon EKS cluster, use the following options to manage the security agent for these resources:

   To enable GuardDuty security agent

   - Enabling automated security agent for Amazon EC2 instance
   - Managing security agent manually for Amazon EC2 resource
   - Managing automated security agent for Fargate (Amazon ECS only)
   - Managing security agent automatically for Amazon EKS resources
   - Managing security agent manually for Amazon EKS cluster

For all existing active member accounts

To enable Runtime Monitoring for existing member accounts in the organization

1. Sign in to the AWS Management Console and open the GuardDuty console at https://console.aws.amazon.com/guardduty/.

   Sign in using the delegated GuardDuty administrator account for the organization.
2. In the navigation pane, choose **Runtime Monitoring**.
3. On the **Runtime Monitoring** page, under the **Configuration** tab, you can view the current status of the Runtime Monitoring configuration.
4. Within the Runtime Monitoring pane, under the **Active member accounts** section, choose **Actions**.

5. From the **Actions** dropdown menu, choose **Enable for all existing active member accounts**.
6. Choose **Confirm**.
7. For GuardDuty to receive the runtime events from one or more resource types – an Amazon EC2 instance, Amazon ECS cluster, or an Amazon EKS cluster, use the following options to manage the security agent for these resources:

   To enable GuardDuty security agent

   - Enabling automated security agent for Amazon EC2 instance
   - Managing security agent manually for Amazon EC2 resource
   - Managing automated security agent for Fargate (Amazon ECS only)
   - Managing security agent automatically for Amazon EKS resources
   - Managing security agent manually for Amazon EKS cluster

> **Note**
>
> It may take up to 24 hours to update the configuration for the member accounts.

Auto-enable Runtime Monitoring for new member accounts only

To enable Runtime Monitoring for new member accounts in your organization

1. Sign in to the AWS Management Console and open the GuardDuty console at https://console.aws.amazon.com/guardduty/.

   Sign in using the designated delegated GuardDuty administrator account of the organization.
2. In the navigation pane, choose **Runtime Monitoring**
3. Under the **Configuration** tab, choose **Edit** in the **Runtime Monitoring configuration** section.
4. Choose **Configure accounts manually**.
5. Select **Automatically enable for new member accounts**.
6. For GuardDuty to receive the runtime events from one or more resource types – an Amazon EC2 instance, Amazon ECS cluster, or an Amazon EKS cluster, use the following options to manage the security agent for these resources:

To enable GuardDuty security agent

- Enabling automated security agent for Amazon EC2 instance
- Managing security agent manually for Amazon EC2 resource
- Managing automated security agent for Fargate (Amazon ECS only)
- Managing security agent automatically for Amazon EKS resources
- Managing security agent manually for Amazon EKS cluster

For selective active member accounts only

To enable Runtime Monitoring for individual active member accounts

1. Open the GuardDuty console at https://console.aws.amazon.com/guardduty/.

   Sign in using the delegated GuardDuty administrator account credentials.
2. In the navigation pane, choose **Accounts**.
3. On the **Accounts** page, review values in the **Runtime Monitoring** and **Manage agent automatically** columns. These values indicate whether Runtime Monitoring and GuardDuty agent management are **Enabled** or **Not enabled** for the corresponding account.
4. From the Accounts table, select the account for which you want to enable Runtime Monitoring. You can choose multiple accounts at a time.
5. Choose **Confirm**.
6. Choose **Edit protection plans**. Choose the appropriate action.
7. Choose **Confirm**.
8. For GuardDuty to receive the runtime events from one or more resource types – an Amazon EC2 instance, Amazon ECS cluster, or an Amazon EKS cluster, use the following options to manage the security agent for these resources:

   To enable GuardDuty security agent

   - Enabling automated security agent for Amazon EC2 instance
   - Managing security agent manually for Amazon EC2 resource
   - Managing automated security agent for Fargate (Amazon ECS only)
   - Managing security agent automatically for Amazon EKS resources

- Managing security agent manually for Amazon EKS cluster

Enabling Runtime Monitoring for a standalone account

A standalone account owns the decision to enable or disable a protection plan in their AWS account in a specific AWS Region.

If your account is associated with a GuardDuty administrator account through AWS Organizations, or by the method of invitation, this section doesn't apply to your account. For more information, see Enabling Runtime Monitoring for multiple-account environments.

After you enable Runtime Monitoring, ensure to install GuardDuty security agent through automated configuration or manual deployment. As a part of completing all the steps listed in the following procedure, make sure to install the security agent.

To enable Runtime Monitoring in standalone account

1. Sign in to the AWS Management Console and open the GuardDuty console at https://console.aws.amazon.com/guardduty/.
2. In the navigation pane, choose **Runtime Monitoring**.
3. Under the **Configuration** tab, choose **Enable** to enable Runtime Monitoring for your account.
4. For GuardDuty to receive the runtime events from one or more resource types – an Amazon EC2 instance, Amazon ECS cluster, or an Amazon EKS cluster, use the following options to manage the security agent for these resources:

   To enable GuardDuty security agent

   - Enabling automated security agent for Amazon EC2 instance
   - Managing security agent manually for Amazon EC2 resource
   - Managing automated security agent for Fargate (Amazon ECS only)
   - Managing security agent automatically for Amazon EKS resources
   - Managing security agent manually for Amazon EKS cluster

Managing GuardDuty security agents

You can manage the GuardDuty security agent for the resource that you want to monitor. If you want to monitor more than one resource type, make sure to manage the GuardDuty agent for that resource.

The following topics will help you with the next steps to manage the security agent.

Contents

- Enabling automated security agent for Amazon EC2 instance
- Managing security agent manually for Amazon EC2 resource
- Managing automated security agent for Fargate (Amazon ECS only)
- Managing security agent automatically for Amazon EKS resources
- Managing security agent manually for Amazon EKS cluster
- Configure GuardDuty security agent (add-on) parameters for Amazon EKS
- Validating VPC endpoint configuration

Enabling automated security agent for Amazon EC2 instance

This section includes steps to enable GuardDuty automated agent for your Amazon EC2 resources in your standalone account or a multiple-account environment.

Before you continue, make sure to follow all the Prerequisites for Amazon EC2 instance support.

If you are migrating from managing the GuardDuty agent manually to enabling GuardDuty automated agent, then before following the steps to enable GuardDuty automated agent, see Migrating from Amazon EC2 manual agent to automated agent.

Enabling GuardDuty agent for Amazon EC2 resources in multiple-account environment

In a multiple-account environments, only the delegated GuardDuty administrator account can enable or disable automated agent configuration for the resource types belonging to the member accounts in their organization. The GuardDuty member accounts can't modify this configuration from their accounts. The delegated GuardDuty administrator account account manages their member accounts using AWS Organizations. For more information about multi-account environments, see [Managing multiple accounts](#).

For delegated GuardDuty administrator account

Configure for all instances

If you chose **Enable for all accounts** for Runtime Monitoring, then choose one of the following options for the delegated GuardDuty administrator account:

- **Option 1**

  Under **Automated agent configuration**, in the **EC2** section, select **Enable for all accounts**.
- **Option 2**
  - Under **Automated agent configuration**, in the **EC2** section, select **Configure accounts manually**.
  - Under **Delegated Administrator (this account)**, choose **Enable**.
- Choose **Save**.

If you chose **Configure accounts manually** for Runtime Monitoring, then perform the following steps:

- Under **Automated agent configuration**, in the **EC2** section, select **Configure accounts manually**.
- Under **Delegated Administrator (this account)**, choose **Enable**.
- Choose **Save**.

Regardless of which option you choose to enable the automated agent configuration for delegated GuardDuty administrator account, you can verify that the SSM association that GuardDuty creates will install and manage the security agent on all the EC2 resources belonging to this account.

1. Open the AWS Systems Manager console at https://console.aws.amazon.com/systems-manager/.
2. Open the **Targets** tab for the SSM association (`GuardDutyRuntimeMonitoring-do-not-delete`). Observe that the **Tag key** appears as **InstanceIds**.

Using inclusion tag in selected instances

To configure GuardDuty agent for selected Amazon EC2 instances

1. Sign in to the AWS Management Console and open the Amazon EC2 console at https://console.aws.amazon.com/ec2/.
2. Add the `GuardDutyManaged:true` tag to the instances that you want GuardDuty to monitor and detect potential threats. For information about adding this tag, see To add a tag to an individual resource.

   Adding this tag will permit GuardDuty to install and manage the security agent for these selected EC2 instances. You **don't** need to enable automated agent configuration explicitly.
3. You can verify that the SSM association that GuardDuty creates will install and manage the security agent only on the EC2 resources that are tagged with the inclusion tags.

   Open the AWS Systems Manager console at https://console.aws.amazon.com/systems-manager/.

   - Open the **Targets** tab for the SSM association that gets created (`GuardDutyRuntimeMonitoring-do-not-delete`). The **Tag key** appears as **tag:GuardDutyManaged**.

Using exclusion tag in selected instances

> **Note**
>
> Ensure that you add the exclusion tag to your Amazon EC2 instances before you launch them. Once you have enabled automated agent configuration for Amazon EC2, any EC2 instance that launches without an exclusion tag will be covered under GuardDuty automated agent configuration.

To configure GuardDuty agent for selected Amazon EC2 instances

1. Sign in to the AWS Management Console and open the Amazon EC2 console at https://console.aws.amazon.com/ec2/.

2. Add the `GuardDutyManaged:false` tag to the instances that you **don't** want GuardDuty to monitor and detect potential threats. For information about adding this tag, see To add a tag to an individual resource.
3. **For the exclusion tags to be available in the instance metadata, perform the following steps:**

   a. Under the **Details** tab of your instance, view the status for **Allow tags in instance metadata**.

      If it is currently **Disabled**, use the following steps to change the status to **Enabled**. Otherwise, skip this step.
   b. Under the **Actions** menu, choose **Instance settings**.
   c. Choose **Allow tags in instance metadata**.
4. After you have added the exclusion tag, perform the same steps as specified in the **Configure for all instances** tab.

You can now assess the runtime Runtime coverage and troubleshooting for Amazon EC2 instance.

Auto-enable for all member accounts

> **Note**
>
> It may take up to 24 hours to update the configuration for the member accounts.

Configure for all instances

The following steps assume that you chose **Enable for all accounts** in the Runtime Monitoring section:

1. Choose **Enable for all accounts** in the **Automated agent configuration** section for **Amazon EC2**.
2. You can verify that the SSM association that GuardDuty creates (`GuardDutyRuntimeMonitoring-do-not-delete`) will install and manage the security agent on all the EC2 resources belonging to this account.

   a. Open the AWS Systems Manager console at https://console.aws.amazon.com/systems-manager/.

b. Open the **Targets** tab for the SSM association. Observe that the **Tag key** appears as **InstanceIds**.

Using inclusion tag in selected instances

To configure GuardDuty agent for selected Amazon EC2 instances

1. Sign in to the AWS Management Console and open the Amazon EC2 console at https://console.aws.amazon.com/ec2/.
2. Add the `GuardDutyManaged`:`true` tag to the EC2 instances that you want GuardDuty to monitor and detect potential threats. For information about adding this tag, see [To add a tag to an individual resource](#).

   Adding this tag will permit GuardDuty to install and manage the security agent for these selected EC2 instances. You **don't** need to enable automated agent configuration explicitly.
3. You can verify that the SSM association that GuardDuty creates will install and manage the security agent on all the EC2 resources belonging to your account.

   a. Open the AWS Systems Manager console at https://console.aws.amazon.com/systems-manager/.

   b. Open the **Targets** tab for the SSM association (`GuardDutyRuntimeMonitoring-do-not-delete`). Observe that the **Tag key** appears as **InstanceIds**.

Using exclusion tag in selected instances

> **Note**
>
> Ensure that you add the exclusion tag to your Amazon EC2 instances before you launch them. Once you have enabled automated agent configuration for Amazon EC2, any EC2 instance that launches without an exclusion tag will be covered under GuardDuty automated agent configuration.

To configure GuardDuty security agent for selected Amazon EC2 instances

1. Sign in to the AWS Management Console and open the Amazon EC2 console at https://console.aws.amazon.com/ec2/.

2. Add the `GuardDutyManaged:false` tag to the instances that you **don't** want GuardDuty to monitor and detect potential threats. For information about adding this tag, see To add a tag to an individual resource.
3. **For the exclusion tags to be available in the instance metadata, perform the following steps:**

   a. Under the **Details** tab of your instance, view the status for **Allow tags in instance metadata**.

      If it is currently **Disabled**, use the following steps to change the status to **Enabled**. Otherwise, skip this step.

   b. Under the **Actions** menu, choose **Instance settings**.

   c. Choose **Allow tags in instance metadata**.
4. After you have added the exclusion tag, perform the same steps as specified in the **Configure for all instances** tab.

You can now assess the runtime Runtime coverage and troubleshooting for Amazon EC2 instance.

Auto-enable for new member accounts only

The delegated GuardDuty administrator account can set the automated agent configuration for Amazon EC2 resource to enable automatically for the new member accounts as they join the organization.

Configure for all instances

The following steps assume that you selected **Automatically enable for new member accounts** under the **Runtime Monitoring** section:

1. In the navigation pane, choose **Runtime Monitoring**.
2. On the **Runtime Monitoring** page, choose **Edit**.
3. Select **Automatically enable for new member accounts**. This step ensures that whenever a new account joins your organization, automated agent configuration for Amazon EC2 will be automatically enabled for their account. Only the delegated GuardDuty administrator account of the organization can modify this selection.
4. Choose **Save**.

When a new member account joins the organization, this configuration will be enabled for them automatically. For GuardDuty to manage the security agent for the Amazon EC2 instances that belong to this new member account, make sure that all the prerequisites For EC2 instance are met.

When an SSM association gets created (`GuardDutyRuntimeMonitoring-do-not-delete`), you can verify that the SSM association will install and manage the security agent on all the EC2 instances belonging to the new member account.

- Open the AWS Systems Manager console at https://console.aws.amazon.com/systems-manager/.
- Open the **Targets** tab for the SSM association. Observe that the **Tag key** appears as **InstanceIds**.

Using inclusion tag in selected instances

To configure GuardDuty security agent for selected instances in your account

1. Sign in to the AWS Management Console and open the Amazon EC2 console at https://console.aws.amazon.com/ec2/.
2. Add the `GuardDutyManaged:true` tag to the instances that you want GuardDuty to monitor and detect potential threats. For information about adding this tag, see To add a tag to an individual resource.

   Adding this tag will permit GuardDuty to install and manage the security agent for these selected instances. You don't need to enable automated agent configuration explicitly.
3. You can verify that the SSM association that GuardDuty creates will install and manage the security agent only on the EC2 resources that are tagged with the inclusion tags.

   a. Open the AWS Systems Manager console at https://console.aws.amazon.com/systems-manager/.

   b. Open the **Targets** tab for the SSM association that gets created. The **Tag key** appears as **tag:GuardDutyManaged**.

Using exclusion tag in selected instances

> **Note**
>
> Ensure that you add the exclusion tag to your Amazon EC2 instances before you launch them. Once you have enabled automated agent configuration for Amazon EC2, any EC2 instance that launches without an exclusion tag will be covered under GuardDuty automated agent configuration.

To configure GuardDuty security agent for specific instances in your standalone account

1. Sign in to the AWS Management Console and open the Amazon EC2 console at https://console.aws.amazon.com/ec2/.
2. Add the `GuardDutyManaged:false` tag to the instances that you **don't** want GuardDuty to monitor and detect potential threats. For information about adding this tag, see To add a tag to an individual resource.
3. **For the exclusion tags to be available in the instance metadata, perform the following steps:**
   a. Under the **Details** tab of your instance, view the status for **Allow tags in instance metadata**.

      If it is currently **Disabled**, use the following steps to change the status to **Enabled**. Otherwise, skip this step.
   b. Under the **Actions** menu, choose **Instance settings**.
   c. Choose **Allow tags in instance metadata**.
4. After you have added the exclusion tag, perform the same steps as specified in the **Configure for all instances** tab.

You can now assess the runtime Runtime coverage and troubleshooting for Amazon EC2 instance.

Selective member accounts only

Configure for all instances

1. On the **Accounts** page, select one or more accounts for which you want to enable **Runtime Monitoring-Automated agent configuration (Amazon EC2)**. Make sure that the accounts that you select in this step already have Runtime Monitoring enabled.
2. From **Edit protection plans**, choose the appropriate option to enable **Runtime Monitoring-Automated agent configuration (Amazon EC2)**.
3. Choose **Confirm**.

Using inclusion tag in selected instances

To configure GuardDuty security agent for selected instances

1. Sign in to the AWS Management Console and open the Amazon EC2 console at https://console.aws.amazon.com/ec2/.
2. Add the `GuardDutyManaged`:`true` tag to the instances that you want GuardDuty to monitor and detect potential threats. For information about adding this tag, see To add a tag to an individual resource.

   Adding this tag will permit GuardDuty to manage the security agent for your tagged Amazon EC2 instances. You don't need to explicitly enable automated agent configuration (**Runtime Monitoring - Automated agent configuration (EC2)**.

Using exclusion tag in selected instances

> **Note**
>
> Ensure that you add the exclusion tag to your Amazon EC2 instances before you launch them. Once you have enabled automated agent configuration for Amazon EC2, any EC2 instance that launches without an exclusion tag will be covered under GuardDuty automated agent configuration.

To configure GuardDuty security agent for selected instances

1. Sign in to the AWS Management Console and open the Amazon EC2 console at https://console.aws.amazon.com/ec2/.
2. Add the `GuardDutyManaged:false` tag to the EC2 instances that you **don't** want GuardDuty to monitor or detect potential threats. For information about adding this tag, see To add a tag to an individual resource.
3. **For the exclusion tags to be available in the instance metadata, perform the following steps:**

    a. Under the **Details** tab of your instance, view the status for **Allow tags in instance metadata**.

       If it is currently **Disabled**, use the following steps to change the status to **Enabled**. Otherwise, skip this step.

    b. Under the **Actions** menu, choose **Instance settings**.

    c. Choose **Allow tags in instance metadata**.
4. After you have added the exclusion tag, perform the same steps as specified in the **Configure for all instances** tab.

You can now assess Runtime coverage and troubleshooting for Amazon EC2 instance.

Enabling GuardDuty automated agent for Amazon EC2 resources in a standalone account

A standalone account owns the decision to enable or disable a protection plan in their AWS account in a specific AWS Region.

If your account is associated with a GuardDuty administrator account through AWS Organizations, or by the method of invitation, this section doesn't apply to your account. For more information, see Enabling Runtime Monitoring for multiple-account environments.

After you enable Runtime Monitoring, ensure to install GuardDuty security agent through automated configuration or manual deployment. As a part of completing all the steps listed in the following procedure, make sure to install the security agent.

Based on your preference to monitor all or selective Amazon EC2 resources, choose a preferred method and follow the steps in the following table.

Configure for all instances

To configure Runtime Monitoring for all instances in your standalone account

1. Sign in to the AWS Management Console and open the GuardDuty console at https://console.aws.amazon.com/guardduty/.
2. In the navigation pane, choose **Runtime Monitoring**.
3. Under the **Configuration** tab, choose **Edit**.
4. In the **EC2** section, choose **Enable**.
5. Choose **Save**.
6. You can verify that the SSM association that GuardDuty creates will install and manage the security agent on all the EC2 resources belonging to your account.

    a. Open the AWS Systems Manager console at https://console.aws.amazon.com/systems-manager/.

    b. Open the **Targets** tab for the SSM association (`GuardDutyRuntimeMonitoring-do-not-delete`). Observe that the **Tag key** appears as **InstanceIds**.

Using inclusion tag in selected instances

To configure GuardDuty security agent for selected Amazon EC2 instances

1. Sign in to the AWS Management Console and open the Amazon EC2 console at https://console.aws.amazon.com/ec2/.
2. Add the `GuardDutyManaged:true` tag to the instances that you want GuardDuty to monitor and detect potential threats. For information about adding this tag, see [To add a tag to an individual resource](#).
3. You can verify that the SSM association that GuardDuty creates will install and manage the security agent only on the EC2 resources that are tagged with the inclusion tags.

    Open the AWS Systems Manager console at https://console.aws.amazon.com/systems-manager/.

    - Open the **Targets** tab for the SSM association that gets created (`GuardDutyRuntimeMonitoring-do-not-delete`). The **Tag key** appears as **tag:GuardDutyManaged**.

Using exclusion tag in selected instances

> **Note**
>
> Ensure that you add the exclusion tag to your Amazon EC2 instances before you launch them. Once you have enabled automated agent configuration for Amazon EC2, any EC2 instance that launches without an exclusion tag will be covered under GuardDuty automated agent configuration.

To configure GuardDuty security agent for selected Amazon EC2 instances

1. Sign in to the AWS Management Console and open the Amazon EC2 console at https://console.aws.amazon.com/ec2/.
2. Add the `GuardDutyManaged:false` tag to the instances that you **don't** want GuardDuty to monitor and detect potential threats. For information about adding this tag, see To add a tag to an individual resource.
3. **For the exclusion tags to be available in the instance metadata, perform the following steps:**

   a. Under the **Details** tab of your instance, view the status for **Allow tags in instance metadata**.

      If it is currently **Disabled**, use the following steps to change the status to **Enabled**. Otherwise, skip this step.

   b. Select the instance for which you want to allow tags.
   c. Under the **Actions** menu, choose **Instance settings**.
   d. Choose **Allow tags in instance metadata**.
   e. Under **Access to tags in instance metadata**, select **Allow**.
   f. Choose **Save**.
4. After you have added the exclusion tag perform the same steps as sepcified in the **Configure for all instances** tab.

You can now assess runtime Runtime coverage and troubleshooting for Amazon EC2 instance.

Migrating from Amazon EC2 manual agent to automated agent

This section applies to your AWS account if you were previously managing the security agent manually and now want to use the GuardDuty automated agent configuration. If this doesn't apply to you, continue with configuring the security agent for your account.

When you enable GuardDuty automated agent, GuardDuty manages the security agent on your behalf. For information about what steps does GuardDuty take, see Use automated agent configuration (recommended).

Clean up resources

Delete SSM association

- Delete any SSM association that you may have created when you were managing the security agent for Amazon EC2 manually. For more information, see Deleting associations.
- This is done so that GuardDuty can take over the management of SSM actions whether you use automated agents at the account level or instance level (by using inclusion or exclusion tags). For more information about what SSM actions can GuardDuty take, see Service-linked role permissions for GuardDuty.
- When you delete an SSM association that was previously created for managing the security agent manually, there might be a brief period of overlap when GuardDuty creates an SSM association for managing the security agent automatically. During this period, you could experience conflicts based on SSM scheduling. For more information, see Amazon EC2 SSM scheduling.

Manage inclusion and exclusion tags for your Amazon EC2 instances

- **Inclusion tags** – When you don't enable GuardDuty automated agent configuration but tag any of your Amazon EC2 instances with an inclusion tag (`GuardDutyManaged`:`true`), GuardDuty creates an SSM association that will install and manage the security agent on the selected EC2 instances. This is an expected behavior that helps you manage the security agent on selected EC2 instances only. For more information, see How Runtime Monitoring works with Amazon EC2 instances.

  To prevent GuardDuty from installing and managing the security agent, remove the inclusion tag from these EC2 instances. For more information, see Add and delete tags in the *Amazon EC2 User Guide*.

- **Exclusion tags –** When you want to enable GuardDuty automated agent configuration for all the EC2 instances in your account, make sure that no EC2 instance is tagged with an exclusion tag (`GuardDutyManaged:false`).

Managing security agent manually for Amazon EC2 resource

This section provides the steps to manually install and update the security agent for your Amazon EC2 resources.

After you enable Runtime Monitoring, you will need to install the GuardDuty security agent manually. To manage the GuardDuty security agent manually, you must first create an Amazon VPC endpoint manually. After this, you can install the security agent so that GuardDuty will start receiving the runtime events from the Amazon EC2 instances. When GuardDuty releases a new agent version for this resource, you can update the agent version in your account.

The following topics include the steps to continuously manage the security agent for your Amazon EC2 resources.

Topics

- Prerequisite – Creating Amazon VPC endpoint manually
- Installing the security agent manually
- Updating the GuardDuty security agent for Amazon EC2 instance manually

Prerequisite – Creating Amazon VPC endpoint manually

Before you can install the GuardDuty security agent, you must create an Amazon Virtual Private Cloud (Amazon VPC) endpoint. This will help GuardDuty receive the runtime events of your Amazon EC2 instances.

> **Note**
>
> There is no additional cost for the usage of the VPC endpoint.

To create a Amazon VPC endpoint

1. Sign in to the AWS Management Console and open the Amazon VPC console at https://console.aws.amazon.com/vpc/.

2. In the navigation pane, under **VPC private cloud**, choose **Endpoints**.
3. Choose **Create Endpoint**.
4. On the **Create endpoint** page, for **Service category**, choose **Other endpoint services**.
5. For **Service name**, enter **`com.amazonaws.us-east-1.guardduty-data`**.

   Make sure to replace *`us-east-1`* with your AWS Region. This must be the same Region as the Amazon EC2 instance that belongs to your AWS account ID.
6. Choose **Verify service**.
7. After the service name is successfully verified, choose the **VPC** where your instance resides. Add the following policy to restrict Amazon VPC endpoint usage to the specified account only. With the organization `Condition` provided below this policy, you can update the following policy to restrict access to your endpoint. To provide the Amazon VPC endpoint support to specific account IDs in your organization, see Organization condition to restrict access to your endpoint.

   JSON

```
{
 "Version": "2012-10-17",
 "Statement": [
  {
   "Action": "*",
   "Resource": "*",
   "Effect": "Allow",
   "Principal": "*"
  },
  {
   "Condition": {
    "StringNotEquals": {
     "aws:PrincipalAccount": "111122223333"
    }
   },
   "Action": "*",
   "Resource": "*",
   "Effect": "Deny",
   "Principal": "*"
  }
 ]
}
```

The `aws:PrincipalAccount` account ID must match the account containing the VPC and VPC endpoint. The following list shows how to share the VPC endpoint with other AWS account IDs:

- To specify multiple accounts to access the VPC endpoint, replace `"aws:PrincipalAccount: "111122223333"` with the following block:

```
"aws:PrincipalAccount": [
          "666666666666",
          "555555555555"
      ]
```

  Make sure to replace the AWS account IDs with the account IDs of those accounts that need to access the VPC endpoint.

- To allow all the members from an organization to access the VPC endpoint, replace `"aws:PrincipalAccount: "111122223333"` with the following line:

```
"aws:PrincipalOrgID": "o-abcdef0123"
```

  Make sure to replace the organization *o-abcdef0123* with your organization ID.

- To restrict accessing a resource by an organization ID, add your `ResourceOrgID` to the policy. For more information, see [aws:ResourceOrgID]() in the *IAM User Guide*.

```
"aws:ResourceOrgID": "o-abcdef0123"
```

8. Under **Additional settings**, choose **Enable DNS name**.
9. Under **Subnets**, choose the subnets in which your instance resides.
10. Under **Security groups**, choose a security group that has the in-bound port 443 enabled from your VPC (or your Amazon EC2 instance). If you don't already have a security group that has an in-bound port 443 enabled, see Create a security group for your VPC in the *Amazon VPC User Guide*.

    If there is an issue while restricting the in-bound permissions to your VPC (or instance), you can the in-bound 443 port from any IP address `(0.0.0.0/0)`. However, GuardDuty recommends using IP addresses that matches the CIDR block for your VPC. For more information, see VPC CIDR blocks in the *Amazon VPC User Guide*.

After you have followed the steps, see Validating VPC endpoint configuration to ensure that the VPC endpoint was set up correctly.

Installing the security agent manually

GuardDuty provides the following two methods to install the GuardDuty security agent on your Amazon EC2 instances. Before proceeding, make sure to follow the steps under Prerequisite – Creating Amazon VPC endpoint manually.

Choose a preferred access method to install the security agent in your Amazon EC2 resources.

- Method 1 - Using AWS Systems Manager – This method requires your Amazon EC2 instance to be AWS Systems Manager managed.
- Method 2 - Using Linux Package Managers – You can use this method whether or not your Amazon EC2 instances are AWS Systems Manager managed. Based on your OS distributions, you can choose an appropriate method to install either RPM scripts or Debian scripts. If you use *Fedora* platform, then you must use this method to install the agent.

Method 1 - Using AWS Systems Manager

To use this method, make sure that your Amazon EC2 instances are AWS Systems Manager managed and then install the agent.

AWS Systems Manager managed Amazon EC2 instance

Use the following steps to make your Amazon EC2 instances AWS Systems Manager managed.

- AWS Systems Manager helps you manage your AWS applications and resources end-to-end and enable secure operations at scale.

  To manage your Amazon EC2 instances with AWS Systems Manager, see Setting up Systems Manager for Amazon EC2 instances in the *AWS Systems Manager User Guide*.
- The following table shows the new GuardDuty managed AWS Systems Manager documents:

Document name	Document type	Purpose
`AmazonGuardDuty-RuntimeMonitoringSsmPlugin`	Distributor	To package the GuardDuty security agent.

Document name	Document type	Purpose
`AmazonGuardDuty-ConfigureRuntimeMonitoringSsmPlugin`	Command	To run installation/un-installation script to install the GuardDuty security agent.

For more information about AWS Systems Manager, see Amazon EC2 Systems Manager Documents in the *AWS Systems Manager User Guide*.

> **For Debian Servers**
>
> The Amazon Machine Images (AMIs) for Debian Server provided by AWS require you to install the AWS Systems Manager agent (SSM agent). You will need to perform an additional step to install the SSM agent to make your Amazon EC2 Debian Server instances SSM managed. For information about steps that you need to take, see Manually installing SSM agent on Debian Server instances in the *AWS Systems Manager User Guide*.

To install the GuardDuty agent for Amazon EC2 instance by using AWS Systems Manager

1. Open the AWS Systems Manager console at https://console.aws.amazon.com/systems-manager/.
2. In the navigation pane, choose **Documents**
3. In **Owned by Amazon**, choose `AmazonGuardDuty-ConfigureRuntimeMonitoringSsmPlugin`.
4. Choose **Run Command**.
5. Enter the following Run Command parameters
   - Action: Choose **Install**.
   - Installation Type: Choose **Install or Uninstall.**
   - Name: `AmazonGuardDuty-RuntimeMonitoringSsmPlugin`
   - Version: If this remains empty, you'll get latest version of the GuardDuty security agent. For more information about the release versions, GuardDuty security agent versions for Amazon EC2 instances.

6. Select the targeted Amazon EC2 instance. You can select one or more Amazon EC2 instances. For more information, see [AWS Systems Manager Running commands from the console] in the *AWS Systems Manager User Guide*
7. Validate if the GuardDuty agent installation is healthy. For more information, see [Validating GuardDuty security agent installation status].

Method 2 - Using Linux Package Managers

With this method, you can install the GuardDuty security agent by running RPM scripts or Debian scripts. Based on the operating systems, you can choose a preferred method:

- Use RPM scripts to install the security agent on OS distributions AL2, AL2023, RedHat, CentOS, or Fedora.
- Use Debian scripts to install the security agent on OS distributions Ubuntu or Debian. For information about supported Ubuntu and Debian OS distributions, see [Validate architectural requirements].

RPM installation

> **Important**
>
> We recommend verifying the GuardDuty security agent RPM signature before installing it on your machine.

1. Verify the GuardDuty security agent RPM signature

   a. **Prepare the template**

      Prepare the commands with appropriate public key, signature of x86_64 RPM, signature of arm64 RPM, and the corresponding access link to the RPM scripts hosted in Amazon S3 buckets. Replace the value of the AWS Region, AWS account ID, and the GuardDuty agent version to access the RPM scripts.

      - **Public key**:

```
s3://694911143906-eu-west-1-guardduty-agent-rpm-artifacts/1.8.0/
publickey.pem
```

- **GuardDuty security agent RPM signature**:

  Signature of x86_64 RPM

  ```
  s3://694911143906-eu-west-1-guardduty-agent-rpm-artifacts/1.8.0/x86_64/
  amazon-guardduty-agent-1.8.0.x86_64.sig
  ```

  Signature of arm64 RPM

  ```
  s3://694911143906-eu-west-1-guardduty-agent-rpm-artifacts/1.8.0/arm64/
  amazon-guardduty-agent-1.8.0.arm64.sig
  ```

- **Access links to the RPM scripts in Amazon S3 bucket**:

  Access link for x86_64 RPM

  ```
  s3://694911143906-eu-west-1-guardduty-agent-rpm-artifacts/1.8.0/x86_64/
  amazon-guardduty-agent-1.8.0.x86_64.rpm
  ```

  Access link for arm64 RPM

  ```
  s3://694911143906-eu-west-1-guardduty-agent-rpm-artifacts/1.8.0/arm64/
  amazon-guardduty-agent-1.8.0.arm64.rpm
  ```

AWS Region	Region name	AWS account ID
`eu-west-1`	Europe (Ireland)	694911143906
`us-east-1`	US East (N. Virginia)	593207742271
`us-west-2`	US West (Oregon)	733349766148
`eu-west-3`	Europe (Paris)	665651866788
`us-east-2`	US East (Ohio)	307168627858
`eu-central-1`	Europe (Frankfurt)	323658145986
`ap-northeast-2`	Asia Pacific (Seoul)	914738172881

eu-north-1	Europe (Stockholm)	591436053604
ap-east-1	Asia Pacific (Hong Kong)	258348409381
me-south-1	Middle East (Bahrain)	536382113932
eu-west-2	Europe (London)	892757235363
ap-northeast-1	Asia Pacific (Tokyo)	533107202818
ap-southeast-1	Asia Pacific (Singapore)	174946120834
ap-south-1	Asia Pacific (Mumbai)	251508486986
ap-southeast-3	Asia Pacific (Jakarta)	510637619217
sa-east-1	South America (São Paulo)	758426053663
ap-northeast-3	Asia Pacific (Osaka)	273192626886
eu-south-1	Europe (Milan)	266869475730
af-south-1	Africa (Cape Town)	197869348890
ap-southeast-2	Asia Pacific (Sydney)	005257825471
me-central-1	Middle East (UAE)	000014521398
us-west-1	US West (N. California)	684579721401
ca-central-1	Canada (Central)	354763396469
ca-west-1	Canada West (Calgary)	339712888787
ap-south-2	Asia Pacific (Hyderabad)	950823858135
eu-south-2	Europe (Spain)	919611009337
eu-central-2	Europe (Zurich)	529164026651
ap-southeast-4	Asia Pacific (Melbourne)	251357961535

ap-southeast-7	Asia Pacific (Thailand)	054037130133
il-central-1	Israel (Tel Aviv)	870907303882
mx-central-1	Mexico (Central)	982081086614
ap-east-2	Asia Pacific (Taipei)	259886477082

b. **Download the template**

In the following command to download appropriate public key, signature of x86_64 RPM, signature of arm64 RPM, and the corresponding access link to the RPM scripts hosted in Amazon S3 buckets, make sure to replace the account ID with the appropriate AWS account ID and the Region with your current Region.

```
aws s3 cp s3://694911143906-eu-west-1-guardduty-agent-rpm-artifacts/1.8.0/
x86_64/amazon-guardduty-agent-1.8.0.x86_64.rpm ./amazon-guardduty-
agent-1.8.0.x86_64.rpm
aws s3 cp s3://694911143906-eu-west-1-guardduty-agent-rpm-artifacts/1.8.0/
x86_64/amazon-guardduty-agent-1.8.0.x86_64.sig ./amazon-guardduty-
agent-1.8.0.x86_64.sig
aws s3 cp s3://694911143906-eu-west-1-guardduty-agent-rpm-artifacts/1.8.0/
publickey.pem ./publickey.pem
```

c. **Import the public key**

Use the following command to import the public key to the database:

```
gpg --import publickey.pem
```

gpg shows import successfully

```
gpg: key 093FF49D: public key "AwsGuardDuty" imported
gpg: Total number processed: 1
gpg:               imported: 1  (RSA: 1)
```

d. **Verify the signature**

Use the following command to verify the signature

```
gpg --verify amazon-guardduty-agent-1.8.0.x86_64.sig amazon-guardduty-
agent-1.8.0.x86_64.rpm
```

If verification passes, you will see a message similar to the result below. You can now proceed to install the GuardDuty security agent using RPM.

Example output:

```
gpg: Signature made Fri 17 Nov 2023 07:58:11 PM UTC using ? key ID 093FF49D
gpg: Good signature from "AwsGuardDuty"
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: 7478 91EF 5378 1334 4456  7603 06C9 06A7 093F F49D
```

If verification fails, it means the signature on RPM has been potentially tampered. You must remove the public key from the database and retry the verification process.

Example:

```
gpg: Signature made Fri 17 Nov 2023 07:58:11 PM UTC using ? key ID 093FF49D
gpg: BAD signature from "AwsGuardDuty"
```

Use the following command to remove the public key from the database:

```
gpg --delete-keys AwsGuardDuty
```

Now, try the verification process again.

2. Connect with SSH from Linux or macOS.
3. Install the GuardDuty security agent by using the following command:

```
sudo rpm -ivh amazon-guardduty-agent-1.8.0.x86_64.rpm
```

4. Validate if the GuardDuty agent installation is healthy. For more information about the steps, see Validating GuardDuty security agent installation status.

Debian installation

> **Important**
>
> We recommend verifying the GuardDuty security agent Debian signature before installing it on your machine.

1. Verify the GuardDuty security agent Debian signature

   a. **Prepare templates for the appropriate public key, signature of amd64 Debian package, signature of arm64 Debian package, and the corresponding access link to the Debian scripts hosted in Amazon S3 buckets**

      In the following templates, replace the value of the AWS Region, AWS account ID, and the GuardDuty agent version to access the Debian package scripts.

      - **Public key**:

        ```
        s3://694911143906-eu-west-1-guardduty-agent-deb-artifacts/1.8.0/
        publickey.pem
        ```

      - **GuardDuty security agent Debian signature**:

        Signature of amd64

        ```
        s3://694911143906-eu-west-1-guardduty-agent-deb-artifacts/1.8.0/amd64/
        amazon-guardduty-agent-1.8.0.amd64.sig
        ```

        Signature of arm64

        ```
        s3://694911143906-eu-west-1-guardduty-agent-deb-artifacts/1.8.0/arm64/
        amazon-guardduty-agent-1.8.0.arm64.sig
        ```

      - **Access links to the Debian scripts in Amazon S3 bucket**:

        Access link for amd64

        ```
        s3://694911143906-eu-west-1-guardduty-agent-deb-artifacts/1.8.0/amd64/
        amazon-guardduty-agent-1.8.0.amd64.deb
        ```

Access link for arm64

```
s3://694911143906-eu-west-1-guardduty-agent-deb-artifacts/1.8.0/arm64/
amazon-guardduty-agent-1.8.0.arm64.deb
```

AWS Region	Region name	AWS account ID
eu-west-1	Europe (Ireland)	694911143906
us-east-1	US East (N. Virginia)	593207742271
us-west-2	US West (Oregon)	733349766148
eu-west-3	Europe (Paris)	665651866788
us-east-2	US East (Ohio)	307168627858
eu-central-1	Europe (Frankfurt)	323658145986
ap-northeast-2	Asia Pacific (Seoul)	914738172881
eu-north-1	Europe (Stockholm)	591436053604
ap-east-1	Asia Pacific (Hong Kong)	258348409381
me-south-1	Middle East (Bahrain)	536382113932
eu-west-2	Europe (London)	892757235363
ap-northeast-1	Asia Pacific (Tokyo)	533107202818
ap-southeast-1	Asia Pacific (Singapore)	174946120834
ap-south-1	Asia Pacific (Mumbai)	251508486986
ap-southeast-3	Asia Pacific (Jakarta)	510637619217
sa-east-1	South America (São Paulo)	758426053663

ap-northeast-3	Asia Pacific (Osaka)	273192626886
eu-south-1	Europe (Milan)	266869475730
af-south-1	Africa (Cape Town)	197869348890
ap-southeast-2	Asia Pacific (Sydney)	005257825471
me-central-1	Middle East (UAE)	000014521398
us-west-1	US West (N. California)	684579721401
ca-central-1	Canada (Central)	354763396469
ca-west-1	Canada West (Calgary)	339712888787
ap-south-2	Asia Pacific (Hyderabad)	950823858135
eu-south-2	Europe (Spain)	919611009337
eu-central-2	Europe (Zurich)	529164026651
ap-southeast-4	Asia Pacific (Melbourne)	251357961535
il-central-1	Israel (Tel Aviv)	870907303882
mx-central-1	Mexico (Central)	982081086614
ap-east-2	Asia Pacific (Taipei)	259886477082

b. **Download the appropriate public key, signature of amd64, signature of arm64, and the corresponding access link to the Debian scripts hosted in Amazon S3 buckets**

In the following commands, replace the account ID with the appropriate AWS account ID, and the Region with your current Region.

```
aws s3 cp s3://694911143906-eu-west-1-guardduty-agent-deb-artifacts/1.8.0/
amd64/amazon-guardduty-agent-1.8.0.amd64.deb ./amazon-guardduty-
agent-1.8.0.amd64.deb
aws s3 cp s3://694911143906-eu-west-1-guardduty-agent-deb-artifacts/1.8.0/
amd64/amazon-guardduty-agent-1.8.0.amd64.sig ./amazon-guardduty-
agent-1.8.0.amd64.sig
```

```
aws s3 cp s3://694911143906-eu-west-1-guardduty-agent-deb-artifacts/1.8.0/
publickey.pem ./publickey.pem
```

c. Import the public key to the database

```
gpg --import publickey.pem
```

gpg shows import successfully

```
gpg: key 093FF49D: public key "AwsGuardDuty" imported
gpg: Total number processed: 1
gpg:               imported: 1  (RSA: 1)
```

d. Verify the signature

```
gpg --verify amazon-guardduty-agent-1.8.0.amd64.sig amazon-guardduty-
agent-1.8.0.amd64.deb
```

After a successful verification, you will see a message similar to the following result:

Example output:

```
gpg: Signature made Fri 17 Nov 2023 07:58:11 PM UTC using ? key ID 093FF49D
gpg: Good signature from "AwsGuardDuty"
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: 7478 91EF 5378 1334 4456  7603 06C9 06A7 093F F49D
```

You can now proceed to install the GuardDuty security agent using Debian.

However, if verification fails, it means the signature in Debian package has been potentially tampered.

Example:

```
gpg: Signature made Fri 17 Nov 2023 07:58:11 PM UTC using ? key ID 093FF49D
gpg: BAD signature from "AwsGuardDuty"
```

Use the following command to remove the public key from the database:

```
gpg --delete-keys AwsGuardDuty
```

Now, retry the verification process.

2. Connect with SSH from Linux or macOS.
3. Install the GuardDuty security agent by using the following command:

```
sudo dpkg -i amazon-guardduty-agent-1.8.0.amd64.deb
```

4. Validate if the GuardDuty agent installation is healthy. For more information about the steps, see Validating GuardDuty security agent installation status.

Out of memory error

If you experience an `out-of-memory` error while installing or updating the GuardDuty security agent for Amazon EC2 manually, see Troubleshooting out of memory error.

Validating GuardDuty security agent installation status

After you have performed the steps to install the GuardDuty security agent, use the following steps to validate the status of the agent:

To validate if the GuardDuty security agent is healthy

1. Connect with SSH from Linux or macOS.
2. Run the following command to check the status of the GuardDuty security agent:

```
sudo systemctl status amazon-guardduty-agent
```

If you want to view the security agent installation logs, they are available under `/var/log/amzn-guardduty-agent/`.

To view the logs, do `sudo journalctl -u amazon-guardduty-agent`.

Updating the GuardDuty security agent for Amazon EC2 instance manually

GuardDuty releases updates to the security agent versions. When you manage the security agent manually, you're responsible to update the agent for your Amazon EC2 instances. For information about new agent versions, see GuardDuty security agent release versions for Amazon EC2 instances. To receive notifications about a new agent version release, see Subscribing to Amazon SNS GuardDuty announcements.

To update the security agent for Amazon EC2 instance manually

The process to update the security agent is the same as installing the security agent. Depending on the method that you used to install the agent, you can perform the steps in Installing the security agent manually for Amazon EC2 instances.

If you use Method 1 - By using AWS Systems Manager, then you can update the security agent by using the **Run command**. Use the agent version to which you want to update.

If you use Method 2 - By using Linux Package Managers, you can use the scripts as specified in the Installing the security agent manually section. The scripts already include the latest agent release version. For information about recently released agent versions, see GuardDuty security agent versions for Amazon EC2 instances.

After you update the security agent, you can check the installation status by looking at the logs. For more information, see Validating GuardDuty security agent installation status.

Managing automated security agent for Fargate (Amazon ECS only)

Runtime Monitoring supports managing the security agent for your Amazon ECS clusters (AWS Fargate) only through GuardDuty. There is no support for managing the security agent manually on Amazon ECS clusters.

Before proceeding with the steps in this section, make sure to follow Prerequisites for AWS Fargate (Amazon ECS only) support.

Based on the Approaches to manage GuardDuty security agent in Amazon ECS-Fargate resources, choose a preferred method to enable GuardDuty automated agent for your resources.

Configuring GuardDuty agent for multi-account environment

In a multiple-account environment, only the delegated GuardDuty administrator account can enable or disable automated agent configuration for the member accounts, and manage automated agent configuration for Amazon ECS clusters that belong to the member accounts in their organization. A GuardDuty member account can't modify this configuration. The delegated GuardDuty administrator account manages their member accounts using AWS Organizations.

For more information about multi-account environments, see Managing multiple accounts in GuardDuty.

Enabling automated agent configuration for delegated GuardDuty administrator account

Manage for all Amazon ECS clusters (account level)

If you chose **Enable for all accounts** for Runtime Monitoring, then you have the following options:

- Choose **Enable for all accounts** in the Automated agent configuration section. GuardDuty will deploy and manage the security agent for all the Amazon ECS tasks that get launched.
- Choose **Configure accounts manually**.

If you chose **Configure accounts manually** in the Runtime Monitoring section, then do the following:

1. Choose **Configure accounts manually** in the Automated agent configuration section.
2. Choose **Enable** in the **delegated GuardDuty administrator account (this account)** section.

Choose **Save**.

When you want GuardDuty to monitor tasks that are part of a service, it requires a new service deployment after you enable Runtime Monitoring. If the last deployment for a specific ECS service was started before you enabled Runtime Monitoring, you can either restart the service, or update the service by using `forceNewDeployment`.

For steps to update the service, see the following resources:

- Updating an Amazon ECS service using the console in the *Amazon Elastic Container Service Developer Guide*.
- UpdateService in the *Amazon Elastic Container Service API Reference*.
- update-service in the *AWS CLI Command Reference*.

Manage for all Amazon ECS clusters but exclude some of the clusters (cluster level)

1. Add a tag to this Amazon ECS cluster with the key-value pair as `GuardDutyManaged`-`false`.

2. Prevent modification of tags, except by the trusted entities. The policy provided in [Prevent tags from being modified except by authorized principles](#) in the *AWS Organizations User Guide* has been modified to be applicable here.

   JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "DenyModifyTagsIfResAuthzTagAndPrinTagDontMatch",
            "Effect": "Deny",
            "Action": [
                "ecs:TagResource",
                "ecs:UntagResource"
            ],
            "Resource": [
                "*"
            ],
            "Condition": {
                "StringNotEquals": {
                    "ecs:ResourceTag/GuardDutyManaged":
 "${aws:PrincipalTag/GuardDutyManaged}",
                    "aws:PrincipalArn": "arn:aws:iam::123456789012:role/
org-admins/iam-admin"
                },
                "Null": {
                    "ecs:ResourceTag/GuardDutyManaged": false
                }
            }
        },
        {
            "Sid": "DenyModifyResAuthzTagIfPrinTagDontMatch",
            "Effect": "Deny",
            "Action": [
                "ecs:TagResource",
                "ecs:UntagResource"
            ],
            "Resource": [
                "*"
            ],
            "Condition": {
```

```
                "StringNotEquals": {
                    "aws:RequestTag/GuardDutyManaged":
 "${aws:PrincipalTag/GuardDutyManaged}",
                    "aws:PrincipalArn": "arn:aws:iam::123456789012:role/
org-admins/iam-admin"
                },
                "ForAnyValue:StringEquals": {
                    "aws:TagKeys": [
                        "GuardDutyManaged"
                    ]
                }
            }
        },
        {
            "Sid": "DenyModifyTagsIfPrinTagNotExists",
            "Effect": "Deny",
            "Action": [
                "ecs:TagResource",
                "ecs:UntagResource"
            ],
            "Resource": [
                "*"
            ],
            "Condition": {
                "StringNotEquals": {
                    "aws:PrincipalArn": "arn:aws:iam::123456789012:role/
org-admins/iam-admin"
                },
                "Null": {
                    "aws:PrincipalTag/GuardDutyManaged": true
                }
            }
        }
    ]
}
```

3. Open the GuardDuty console at https://console.aws.amazon.com/guardduty/.
4. In the navigation pane, choose **Runtime Monitoring**.
5. **Note**

   Always add the exclusion tag to your Amazon ECS clusters before enabling Automated agent configuration for your account; otherwise the GuardDuty sidecar container will be attached to all the containers in the Amazon ECS tasks that get launched.

Under the **Configuration** tab, choose **Enable** in the **Automated agent configuration**.

For the Amazon ECS clusters that have not been excluded, GuardDuty will manage the deployment of the security agent in the sidecar container.

6. Choose **Save**.
7. When you want GuardDuty to monitor tasks that are part of a service, it requires a new service deployment after you enable Runtime Monitoring. If the last deployment for a specific ECS service was started before you enabled Runtime Monitoring, you can either restart the service, or update the service by using `forceNewDeployment`.

   For steps to update the service, see the following resources:

   - Updating an Amazon ECS service using the console in the *Amazon Elastic Container Service Developer Guide*.
   - UpdateService in the *Amazon Elastic Container Service API Reference*.
   - update-service in the *AWS CLI Command Reference*.

Manage for selective (inclusion only) Amazon ECS clusters (cluster level)

1. Add a tag to an Amazon ECS cluster for which you want to include all of the tasks. The key-value pair must be `GuardDutyManaged`-`true`.
2. Prevent modification of these tags, except by trusted entities. The policy provided in Prevent tags from being modified except by authorized principles in the *AWS Organizations User Guide* has been modified to be applicable here.

   JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "DenyModifyTagsIfResAuthzTagAndPrinTagDontMatch",
            "Effect": "Deny",
```

```
            "Action": [
                "ecs:TagResource",
                "ecs:UntagResource"
            ],
            "Resource": [
                "*"
            ],
            "Condition": {
                "StringNotEquals": {
                    "ecs:ResourceTag/GuardDutyManaged":
 "${aws:PrincipalTag/GuardDutyManaged}",
                    "aws:PrincipalArn": "arn:aws:iam::123456789012:role/
org-admins/iam-admin"
                },
                "Null": {
                    "ecs:ResourceTag/GuardDutyManaged": false
                }
            }
        },
        {
            "Sid": "DenyModifyResAuthzTagIfPrinTagDontMatch",
            "Effect": "Deny",
            "Action": [
                "ecs:TagResource",
                "ecs:UntagResource"
            ],
            "Resource": [
                "*"
            ],
            "Condition": {
                "StringNotEquals": {
                    "aws:RequestTag/GuardDutyManaged":
 "${aws:PrincipalTag/GuardDutyManaged}",
                    "aws:PrincipalArn": "arn:aws:iam::123456789012:role/
org-admins/iam-admin"
                },
                "ForAnyValue:StringEquals": {
                    "aws:TagKeys": [
                        "GuardDutyManaged"
                    ]
                }
            }
        },
        {
```

```
            "Sid": "DenyModifyTagsIfPrinTagNotExists",
            "Effect": "Deny",
            "Action": [
                "ecs:TagResource",
                "ecs:UntagResource"
            ],
            "Resource": [
                "*"
            ],
            "Condition": {
                "StringNotEquals": {
                    "aws:PrincipalArn": "arn:aws:iam::123456789012:role/
org-admins/iam-admin"
                },
                "Null": {
                    "aws:PrincipalTag/GuardDutyManaged": true
                }
            }
        }
    ]
}
```

Note

When using inclusion tags for your Amazon ECS clusters, you don't need to enable GuardDuty agent through automated agent congifuration explicitly.

3. When you want GuardDuty to monitor tasks that are part of a service, it requires a new service deployment after you enable Runtime Monitoring. If the last deployment for a specific ECS service was started before you enabled Runtime Monitoring, you can either restart the service, or update the service by using `forceNewDeployment`.

   For steps to update the service, see the following resources:

   - Updating an Amazon ECS service using the console in the *Amazon Elastic Container Service Developer Guide*.
   - UpdateService in the *Amazon Elastic Container Service API Reference*.
   - update-service in the *AWS CLI Command Reference*.

Auto-enable for all member accounts

Manage for all Amazon ECS clusters (account level)

The following steps assume that you chose **Enable for all accounts** in the Runtime Monitoring section.

1. Choose **Enable for all accounts** in the Automated agent configuration section. GuardDuty will deploy and manage the security agent for all the Amazon ECS tasks that get launched.
2. Choose **Save**.
3. When you want GuardDuty to monitor tasks that are part of a service, it requires a new service deployment after you enable Runtime Monitoring. If the last deployment for a specific ECS service was started before you enabled Runtime Monitoring, you can either restart the service, or update the service by using `forceNewDeployment`.

   For steps to update the service, see the following resources:

   - Updating an Amazon ECS service using the console in the *Amazon Elastic Container Service Developer Guide*.
   - UpdateService in the *Amazon Elastic Container Service API Reference*.
   - update-service in the *AWS CLI Command Reference*.

Manage for all Amazon ECS clusters but exclude some of the clusters (cluster level)

1. Add a tag to this Amazon ECS cluster with the key-value pair as `GuardDutyManaged`-`false`.
2. Prevent modification of tags, except by the trusted entities. The policy provided in Prevent tags from being modified except by authorized principles in the *AWS Organizations User Guide* has been modified to be applicable here.

   JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "DenyModifyTagsIfResAuthzTagAndPrinTagDontMatch",
            "Effect": "Deny",
```

```
            "Action": [
                "ecs:TagResource",
                "ecs:UntagResource"
            ],
            "Resource": [
                "*"
            ],
            "Condition": {
                "StringNotEquals": {
                    "ecs:ResourceTag/GuardDutyManaged":
 "${aws:PrincipalTag/GuardDutyManaged}",
                    "aws:PrincipalArn": "arn:aws:iam::123456789012:role/
org-admins/iam-admin"
                },
                "Null": {
                    "ecs:ResourceTag/GuardDutyManaged": false
                }
            }
        },
        {
            "Sid": "DenyModifyResAuthzTagIfPrinTagDontMatch",
            "Effect": "Deny",
            "Action": [
                "ecs:TagResource",
                "ecs:UntagResource"
            ],
            "Resource": [
                "*"
            ],
            "Condition": {
                "StringNotEquals": {
                    "aws:RequestTag/GuardDutyManaged":
 "${aws:PrincipalTag/GuardDutyManaged}",
                    "aws:PrincipalArn": "arn:aws:iam::123456789012:role/
org-admins/iam-admin"
                },
                "ForAnyValue:StringEquals": {
                    "aws:TagKeys": [
                        "GuardDutyManaged"
                    ]
                }
            }
        },
        {
```

```
            "Sid": "DenyModifyTagsIfPrinTagNotExists",
            "Effect": "Deny",
            "Action": [
                "ecs:TagResource",
                "ecs:UntagResource"
            ],
            "Resource": [
                "*"
            ],
            "Condition": {
                "StringNotEquals": {
                    "aws:PrincipalArn": "arn:aws:iam::123456789012:role/
org-admins/iam-admin"
                },
                "Null": {
                    "aws:PrincipalTag/GuardDutyManaged": true
                }
            }
        }
    ]
}
```

3. Open the GuardDuty console at https://console.aws.amazon.com/guardduty/.
4. In the navigation pane, choose **Runtime Monitoring**.
5. **Note**

   Always add the exclusion tag to your Amazon ECS clusters before enabling Automated agent configuration for your account; otherwise the GuardDuty sidecar container will be attached to all the containers in the Amazon ECS tasks that get launched.

   Under the **Configuration** tab, choose **Edit**.
6. Choose **Enable for all accounts** in the **Automated agent configuration** section

   For the Amazon ECS clusters that have not been excluded, GuardDuty will manage the deployment of the security agent in the sidecar container.
7. Choose **Save**.
8. When you want GuardDuty to monitor tasks that are part of a service, it requires a new service deployment after you enable Runtime Monitoring. If the last deployment for a

specific ECS service was started before you enabled Runtime Monitoring, you can either restart the service, or update the service by using `forceNewDeployment`.

For steps to update the service, see the following resources:

- [Updating an Amazon ECS service using the console](#) in the *Amazon Elastic Container Service Developer Guide*.
- [UpdateService](#) in the *Amazon Elastic Container Service API Reference*.
- [update-service](#) in the *AWS CLI Command Reference*.

Manage for selective (inclusion-only) Amazon ECS clusters (cluster level)

Regardless of how you choose to enable Runtime Monitoring, the following steps will help you monitor selective Amazon ECS Fargate tasks for all of the member accounts in your organization.

1. Do not enable any configuration in the Automated agent configuration section. Keep the Runtime Monitoring configuration the same as you selected in the previous step.
2. Choose **Save**.
3. Prevent modification of these tags, except by trusted entities. The policy provided in [Prevent tags from being modified except by authorized principles](#) in the *AWS Organizations User Guide* has been modified to be applicable here.

   JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "DenyModifyTagsIfResAuthzTagAndPrinTagDontMatch",
            "Effect": "Deny",
            "Action": [
                "ecs:TagResource",
                "ecs:UntagResource"
            ],
            "Resource": [
                "*"
            ],
            "Condition": {
```

```
                "StringNotEquals": {
                    "ecs:ResourceTag/GuardDutyManaged":
 "${aws:PrincipalTag/GuardDutyManaged}",
                    "aws:PrincipalArn": "arn:aws:iam::123456789012:role/
org-admins/iam-admin"
                },
                "Null": {
                    "ecs:ResourceTag/GuardDutyManaged": false
                }
            }
        },
        {
            "Sid": "DenyModifyResAuthzTagIfPrinTagDontMatch",
            "Effect": "Deny",
            "Action": [
                "ecs:TagResource",
                "ecs:UntagResource"
            ],
            "Resource": [
                "*"
            ],
            "Condition": {
                "StringNotEquals": {
                    "aws:RequestTag/GuardDutyManaged":
 "${aws:PrincipalTag/GuardDutyManaged}",
                    "aws:PrincipalArn": "arn:aws:iam::123456789012:role/
org-admins/iam-admin"
                },
                "ForAnyValue:StringEquals": {
                    "aws:TagKeys": [
                        "GuardDutyManaged"
                    ]
                }
            }
        },
        {
            "Sid": "DenyModifyTagsIfPrinTagNotExists",
            "Effect": "Deny",
            "Action": [
                "ecs:TagResource",
                "ecs:UntagResource"
            ],
            "Resource": [
                "*"
```

```
            ],
            "Condition": {
                "StringNotEquals": {
                    "aws:PrincipalArn": "arn:aws:iam::123456789012:role/
org-admins/iam-admin"
                },
                "Null": {
                    "aws:PrincipalTag/GuardDutyManaged": true
                }
            }
        }
    ]
}
```

> **Note**
>
> When using inclusion tags for your Amazon ECS clusters, you don't need to enable **GuardDuty agent auto-management** explicitly.

4. When you want GuardDuty to monitor tasks that are part of a service, it requires a new service deployment after you enable Runtime Monitoring. If the last deployment for a specific ECS service was started before you enabled Runtime Monitoring, you can either restart the service, or update the service by using `forceNewDeployment`.

   For steps to update the service, see the following resources:

   - Updating an Amazon ECS service using the console in the *Amazon Elastic Container Service Developer Guide*.
   - UpdateService in the *Amazon Elastic Container Service API Reference*.
   - update-service in the *AWS CLI Command Reference*.

Enabling automated agent configuration for existing active member accounts

Manage for all Amazon ECS clusters (account level)

1. On the Runtime Monitoring page, under the **Configuration** tab, you can view the current status of Automated agent configuration.

2. Within the Automated agent configuration pane, under the **Active member accounts** section, choose **Actions**.
3. From **Actions**, choose **Enable for all existing active member accounts**.
4. Choose **Confirm**.
5. When you want GuardDuty to monitor tasks that are part of a service, it requires a new service deployment after you enable Runtime Monitoring. If the last deployment for a specific ECS service was started before you enabled Runtime Monitoring, you can either restart the service, or update the service by using `forceNewDeployment`.

   For steps to update the service, see the following resources:

   - Updating an Amazon ECS service using the console in the *Amazon Elastic Container Service Developer Guide*.
   - UpdateService in the *Amazon Elastic Container Service API Reference*.
   - update-service in the *AWS CLI Command Reference*.

Manage for all Amazon ECS clusters but exclude some of the clusters (cluster level)

1. Add a tag to this Amazon ECS cluster with the key-value pair as `GuardDutyManaged`-`false`.
2. Prevent modification of tags, except by the trusted entities. The policy provided in Prevent tags from being modified except by authorized principles in the *AWS Organizations User Guide* has been modified to be applicable here.

   JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "DenyModifyTagsIfResAuthzTagAndPrinTagDontMatch",
            "Effect": "Deny",
            "Action": [
                "ecs:TagResource",
                "ecs:UntagResource"
            ],
            "Resource": [
                "*"
```

```
            ],
            "Condition": {
                "StringNotEquals": {
                    "ecs:ResourceTag/GuardDutyManaged":
 "${aws:PrincipalTag/GuardDutyManaged}",
                    "aws:PrincipalArn": "arn:aws:iam::123456789012:role/
org-admins/iam-admin"
                },
                "Null": {
                    "ecs:ResourceTag/GuardDutyManaged": false
                }
            }
        },
        {
            "Sid": "DenyModifyResAuthzTagIfPrinTagDontMatch",
            "Effect": "Deny",
            "Action": [
                "ecs:TagResource",
                "ecs:UntagResource"
            ],
            "Resource": [
                "*"
            ],
            "Condition": {
                "StringNotEquals": {
                    "aws:RequestTag/GuardDutyManaged":
 "${aws:PrincipalTag/GuardDutyManaged}",
                    "aws:PrincipalArn": "arn:aws:iam::123456789012:role/
org-admins/iam-admin"
                },
                "ForAnyValue:StringEquals": {
                    "aws:TagKeys": [
                        "GuardDutyManaged"
                    ]
                }
            }
        },
        {
            "Sid": "DenyModifyTagsIfPrinTagNotExists",
            "Effect": "Deny",
            "Action": [
                "ecs:TagResource",
                "ecs:UntagResource"
            ],
```

```
            "Resource": [
                "*"
            ],
            "Condition": {
                "StringNotEquals": {
                    "aws:PrincipalArn": "arn:aws:iam::123456789012:role/
org-admins/iam-admin"
                },
                "Null": {
                    "aws:PrincipalTag/GuardDutyManaged": true
                }
            }
        }
    ]
}
```

3. Open the GuardDuty console at https://console.aws.amazon.com/guardduty/.
4. In the navigation pane, choose **Runtime Monitoring**.
5. > **Note**
   >
   > Always add the exclusion tag to your Amazon ECS clusters before enabling Automated agent configuration for your account; otherwise the GuardDuty sidecar container will be attached to all the containers in the Amazon ECS tasks that get launched.

   Under the **Configuration** tab, in the Automated agent configuration section, under **Active member accounts**, choose **Actions**.
6. From **Actions**, choose **Enable for all active member accounts**.

   For the Amazon ECS clusters that have not been excluded, GuardDuty will manage the deployment of the security agent in the sidecar container.
7. Choose **Confirm**.
8. When you want GuardDuty to monitor tasks that are part of a service, it requires a new service deployment after you enable Runtime Monitoring. If the last deployment for a specific ECS service was started before you enabled Runtime Monitoring, you can either restart the service, or update the service by using `forceNewDeployment`.

   For steps to update the service, see the following resources:

- Updating an Amazon ECS service using the console in the *Amazon Elastic Container Service Developer Guide*.
- UpdateService in the *Amazon Elastic Container Service API Reference*.
- update-service in the *AWS CLI Command Reference*.

Manage for selective (inclusion only) Amazon ECS clusters (cluster level)

1. Add a tag to an Amazon ECS cluster for which you want to include all of the tasks. The key-value pair must be `GuardDutyManaged`-`true`.
2. Prevent modification of these tags, except by trusted entities. The policy provided in Prevent tags from being modified except by authorized principles in the *AWS Organizations User Guide* has been modified to be applicable here.

   JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "DenyModifyTagsIfResAuthzTagAndPrinTagDontMatch",
            "Effect": "Deny",
            "Action": [
                "ecs:TagResource",
                "ecs:UntagResource"
            ],
            "Resource": [
                "*"
            ],
            "Condition": {
                "StringNotEquals": {
                    "ecs:ResourceTag/GuardDutyManaged":
 "${aws:PrincipalTag/GuardDutyManaged}",
                    "aws:PrincipalArn": "arn:aws:iam::123456789012:role/
org-admins/iam-admin"
                },
                "Null": {
                    "ecs:ResourceTag/GuardDutyManaged": false
                }
            }
```

```
        },
        {
            "Sid": "DenyModifyResAuthzTagIfPrinTagDontMatch",
            "Effect": "Deny",
            "Action": [
                "ecs:TagResource",
                "ecs:UntagResource"
            ],
            "Resource": [
                "*"
            ],
            "Condition": {
                "StringNotEquals": {
                    "aws:RequestTag/GuardDutyManaged":
 "${aws:PrincipalTag/GuardDutyManaged}",
                    "aws:PrincipalArn": "arn:aws:iam::123456789012:role/
org-admins/iam-admin"
                },
                "ForAnyValue:StringEquals": {
                    "aws:TagKeys": [
                        "GuardDutyManaged"
                    ]
                }
            }
        },
        {
            "Sid": "DenyModifyTagsIfPrinTagNotExists",
            "Effect": "Deny",
            "Action": [
                "ecs:TagResource",
                "ecs:UntagResource"
            ],
            "Resource": [
                "*"
            ],
            "Condition": {
                "StringNotEquals": {
                    "aws:PrincipalArn": "arn:aws:iam::123456789012:role/
org-admins/iam-admin"
                },
                "Null": {
                    "aws:PrincipalTag/GuardDutyManaged": true
                }
            }
```

```
        }
    ]
}
```

> **Note**
>
> When using inclusion tags for your Amazon ECS clusters, you don't need to enable **Automated agent configuration** explicitly.

3. When you want GuardDuty to monitor tasks that are part of a service, it requires a new service deployment after you enable Runtime Monitoring. If the last deployment for a specific ECS service was started before you enabled Runtime Monitoring, you can either restart the service, or update the service by using `forceNewDeployment`.

   For steps to update the service, see the following resources:

   - Updating an Amazon ECS service using the console in the *Amazon Elastic Container Service Developer Guide*.
   - UpdateService in the *Amazon Elastic Container Service API Reference*.
   - update-service in the *AWS CLI Command Reference*.

Auto-enable Automated agent configuration for new members

Manage for all Amazon ECS clusters (account level)

1. On the Runtime Monitoring page, choose **Edit** to update the existing configuration.
2. In the Automated agent configuration section, select **Automatically enable for new member accounts**.
3. Choose **Save**.
4. When you want GuardDuty to monitor tasks that are part of a service, it requires a new service deployment after you enable Runtime Monitoring. If the last deployment for a specific ECS service was started before you enabled Runtime Monitoring, you can either restart the service, or update the service by using `forceNewDeployment`.

   For steps to update the service, see the following resources:

- Updating an Amazon ECS service using the console in the *Amazon Elastic Container Service Developer Guide*.
- UpdateService in the *Amazon Elastic Container Service API Reference*.
- update-service in the *AWS CLI Command Reference*.

Manage for all Amazon ECS clusters but exclude some of the clusters (cluster level)

1. Add a tag to this Amazon ECS cluster with the key-value pair as `GuardDutyManaged-false`.
2. Prevent modification of tags, except by the trusted entities. The policy provided in Prevent tags from being modified except by authorized principles in the *AWS Organizations User Guide* has been modified to be applicable here.

   JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "DenyModifyTagsIfResAuthzTagAndPrinTagDontMatch",
            "Effect": "Deny",
            "Action": [
                "ecs:TagResource",
                "ecs:UntagResource"
            ],
            "Resource": [
                "*"
            ],
            "Condition": {
                "StringNotEquals": {
                    "ecs:ResourceTag/GuardDutyManaged":
 "${aws:PrincipalTag/GuardDutyManaged}",
                    "aws:PrincipalArn": "arn:aws:iam::123456789012:role/
org-admins/iam-admin"
                },
                "Null": {
                    "ecs:ResourceTag/GuardDutyManaged": false
                }
            }
```

```
        },
        {
            "Sid": "DenyModifyResAuthzTagIfPrinTagDontMatch",
            "Effect": "Deny",
            "Action": [
                "ecs:TagResource",
                "ecs:UntagResource"
            ],
            "Resource": [
                "*"
            ],
            "Condition": {
                "StringNotEquals": {
                    "aws:RequestTag/GuardDutyManaged":
 "${aws:PrincipalTag/GuardDutyManaged}",
                    "aws:PrincipalArn": "arn:aws:iam::123456789012:role/
org-admins/iam-admin"
                },
                "ForAnyValue:StringEquals": {
                    "aws:TagKeys": [
                        "GuardDutyManaged"
                    ]
                }
            }
        },
        {
            "Sid": "DenyModifyTagsIfPrinTagNotExists",
            "Effect": "Deny",
            "Action": [
                "ecs:TagResource",
                "ecs:UntagResource"
            ],
            "Resource": [
                "*"
            ],
            "Condition": {
                "StringNotEquals": {
                    "aws:PrincipalArn": "arn:aws:iam::123456789012:role/
org-admins/iam-admin"
                },
                "Null": {
                    "aws:PrincipalTag/GuardDutyManaged": true
                }
            }
```

```
        }
    ]
}
```

3. Open the GuardDuty console at https://console.aws.amazon.com/guardduty/.
4. In the navigation pane, choose **Runtime Monitoring**.
5. > **Note**
   >
   > Always add the exclusion tag to your Amazon ECS clusters before enabling Automated agent configuration for your account; otherwise the GuardDuty sidecar container will be attached to all the containers in the Amazon ECS tasks that get launched.

   Under the **Configuration** tab, select **Automatically enable for new member accounts** in the **Automated agent configuration** section.

   For the Amazon ECS clusters that have not been excluded, GuardDuty will manage the deployment of the security agent in the sidecar container.
6. Choose **Save**.
7. When you want GuardDuty to monitor tasks that are part of a service, it requires a new service deployment after you enable Runtime Monitoring. If the last deployment for a specific ECS service was started before you enabled Runtime Monitoring, you can either restart the service, or update the service by using `forceNewDeployment`.

   For steps to update the service, see the following resources:

   - Updating an Amazon ECS service using the console in the *Amazon Elastic Container Service Developer Guide*.
   - UpdateService in the *Amazon Elastic Container Service API Reference*.
   - update-service in the *AWS CLI Command Reference*.

Manage for selective (inclusion only) Amazon ECS clusters (cluster level)

1. Add a tag to an Amazon ECS cluster for which you want to include all of the tasks. The key-value pair must be `GuardDutyManaged`-`true`.

2. Prevent modification of these tags, except by trusted entities. The policy provided in [Prevent tags from being modified except by authorized principles](#) in the *AWS Organizations User Guide* has been modified to be applicable here.

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "DenyModifyTagsIfResAuthzTagAndPrinTagDontMatch",
            "Effect": "Deny",
            "Action": [
                "ecs:TagResource",
                "ecs:UntagResource"
            ],
            "Resource": [
                "*"
            ],
            "Condition": {
                "StringNotEquals": {
                    "ecs:ResourceTag/GuardDutyManaged":
 "${aws:PrincipalTag/GuardDutyManaged}",
                    "aws:PrincipalArn": "arn:aws:iam::123456789012:role/
org-admins/iam-admin"
                },
                "Null": {
                    "ecs:ResourceTag/GuardDutyManaged": false
                }
            }
        },
        {
            "Sid": "DenyModifyResAuthzTagIfPrinTagDontMatch",
            "Effect": "Deny",
            "Action": [
                "ecs:TagResource",
                "ecs:UntagResource"
            ],
            "Resource": [
                "*"
            ],
            "Condition": {
```

```
                "StringNotEquals": {
                    "aws:RequestTag/GuardDutyManaged":
 "${aws:PrincipalTag/GuardDutyManaged}",
                    "aws:PrincipalArn": "arn:aws:iam::123456789012:role/
org-admins/iam-admin"
                },
                "ForAnyValue:StringEquals": {
                    "aws:TagKeys": [
                        "GuardDutyManaged"
                    ]
                }
            }
        },
        {
            "Sid": "DenyModifyTagsIfPrinTagNotExists",
            "Effect": "Deny",
            "Action": [
                "ecs:TagResource",
                "ecs:UntagResource"
            ],
            "Resource": [
                "*"
            ],
            "Condition": {
                "StringNotEquals": {
                    "aws:PrincipalArn": "arn:aws:iam::123456789012:role/
org-admins/iam-admin"
                },
                "Null": {
                    "aws:PrincipalTag/GuardDutyManaged": true
                }
            }
        }
    ]
}
```

Note

When using inclusion tags for your Amazon ECS clusters, you don't need to enable **Automated agent configuration** explicitly.

3. When you want GuardDuty to monitor tasks that are part of a service, it requires a new service deployment after you enable Runtime Monitoring. If the last deployment for a specific ECS service was started before you enabled Runtime Monitoring, you can either restart the service, or update the service by using `forceNewDeployment`.

   For steps to update the service, see the following resources:

   - Updating an Amazon ECS service using the console in the *Amazon Elastic Container Service Developer Guide*.
   - UpdateService in the *Amazon Elastic Container Service API Reference*.
   - update-service in the *AWS CLI Command Reference*.

Enabling Automated agent configuration for active member accounts selectively

Manage for all Amazon ECS (account level)

1. On the Accounts page, select the accounts for which you want to enable Runtime Monitoring-Automated agent configuration (ECS-Fargate). You can select multiple accounts. Make sure that the accounts that you select in this step are already enabled with Runtime Monitoring.
2. From **Edit protection plans**, choose the appropriate option to enable **Runtime Monitoring-Automated agent configuration (ECS-Fargate)**.
3. Choose **Confirm**.
4. When you want GuardDuty to monitor tasks that are part of a service, it requires a new service deployment after you enable Runtime Monitoring. If the last deployment for a specific ECS service was started before you enabled Runtime Monitoring, you can either restart the service, or update the service by using `forceNewDeployment`.

   For steps to update the service, see the following resources:

   - Updating an Amazon ECS service using the console in the *Amazon Elastic Container Service Developer Guide*.
   - UpdateService in the *Amazon Elastic Container Service API Reference*.
   - update-service in the *AWS CLI Command Reference*.

Manage for all Amazon ECS clusters but exclude some of the clusters (cluster level)

1. Add a tag to this Amazon ECS cluster with the key-value pair as `GuardDutyManaged`-`false`.
2. Prevent modification of tags, except by the trusted entities. The policy provided in [Prevent tags from being modified except by authorized principles]() in the *AWS Organizations User Guide* has been modified to be applicable here.

   JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "DenyModifyTagsIfResAuthzTagAndPrinTagDontMatch",
            "Effect": "Deny",
            "Action": [
                "ecs:TagResource",
                "ecs:UntagResource"
            ],
            "Resource": [
                "*"
            ],
            "Condition": {
                "StringNotEquals": {
                    "ecs:ResourceTag/GuardDutyManaged":
 "${aws:PrincipalTag/GuardDutyManaged}",
                    "aws:PrincipalArn": "arn:aws:iam::123456789012:role/
org-admins/iam-admin"
                },
                "Null": {
                    "ecs:ResourceTag/GuardDutyManaged": false
                }
            }
        },
        {
            "Sid": "DenyModifyResAuthzTagIfPrinTagDontMatch",
            "Effect": "Deny",
            "Action": [
                "ecs:TagResource",
                "ecs:UntagResource"
            ],
```

```
            "Resource": [
                "*"
            ],
            "Condition": {
                "StringNotEquals": {
                    "aws:RequestTag/GuardDutyManaged":
 "${aws:PrincipalTag/GuardDutyManaged}",
                    "aws:PrincipalArn": "arn:aws:iam::123456789012:role/
org-admins/iam-admin"
                },
                "ForAnyValue:StringEquals": {
                    "aws:TagKeys": [
                        "GuardDutyManaged"
                    ]
                }
            }
        },
        {
            "Sid": "DenyModifyTagsIfPrinTagNotExists",
            "Effect": "Deny",
            "Action": [
                "ecs:TagResource",
                "ecs:UntagResource"
            ],
            "Resource": [
                "*"
            ],
            "Condition": {
                "StringNotEquals": {
                    "aws:PrincipalArn": "arn:aws:iam::123456789012:role/
org-admins/iam-admin"
                },
                "Null": {
                    "aws:PrincipalTag/GuardDutyManaged": true
                }
            }
        }
    ]
}
```

3. Open the GuardDuty console at https://console.aws.amazon.com/guardduty/.

4. In the navigation pane, choose **Runtime Monitoring**.

5. > **Note**
   >
   > Always add the exclusion tag to your Amazon ECS clusters before enabling GuardDuty agent auto-management for your account; otherwise the GuardDuty sidecar container will be attached to all the containers in the Amazon ECS tasks that get launched.

   On the Accounts page, select the accounts for which you want to enable Runtime Monitoring-Automated agent configuration (ECS-Fargate). You can select multiple accounts. Make sure that the accounts that you select in this step are already enabled with Runtime Monitoring.

   For the Amazon ECS clusters that have not been excluded, GuardDuty will manage the deployment of the security agent in the sidecar container.

6. From **Edit protection plans**, choose the appropriate option to enable **Runtime Monitoring-Automated agent configuration (ECS-Fargate)**.
7. Choose **Save**.
8. When you want GuardDuty to monitor tasks that are part of a service, it requires a new service deployment after you enable Runtime Monitoring. If the last deployment for a specific ECS service was started before you enabled Runtime Monitoring, you can either restart the service, or update the service by using `forceNewDeployment`.

   For steps to update the service, see the following resources:

   - Updating an Amazon ECS service using the console in the *Amazon Elastic Container Service Developer Guide*.
   - UpdateService in the *Amazon Elastic Container Service API Reference*.
   - update-service in the *AWS CLI Command Reference*.

Manage for selective (inclusion only) Amazon ECS clusters (cluster level)

1. Make sure you don't enable **Automated agent configuration** (or **Runtime Monitoring-Automated agent configuration (ECS-Fargate)**) for the selected accounts that have the Amazon ECS clusters that you want to monitor.

2. Add a tag to an Amazon ECS cluster for which you want to include all of the tasks. The key-value pair must be `GuardDutyManaged-true`.
3. Prevent modification of these tags, except by trusted entities. The policy provided in [Prevent tags from being modified except by authorized principles]() in the *AWS Organizations User Guide* has been modified to be applicable here.

   JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "DenyModifyTagsIfResAuthzTagAndPrinTagDontMatch",
            "Effect": "Deny",
            "Action": [
                "ecs:TagResource",
                "ecs:UntagResource"
            ],
            "Resource": [
                "*"
            ],
            "Condition": {
                "StringNotEquals": {
                    "ecs:ResourceTag/GuardDutyManaged":
 "${aws:PrincipalTag/GuardDutyManaged}",
                    "aws:PrincipalArn": "arn:aws:iam::123456789012:role/
org-admins/iam-admin"
                },
                "Null": {
                    "ecs:ResourceTag/GuardDutyManaged": false
                }
            }
        },
        {
            "Sid": "DenyModifyResAuthzTagIfPrinTagDontMatch",
            "Effect": "Deny",
            "Action": [
                "ecs:TagResource",
                "ecs:UntagResource"
            ],
            "Resource": [
                "*"
```

```
            ],
            "Condition": {
                "StringNotEquals": {
                    "aws:RequestTag/GuardDutyManaged":
 "${aws:PrincipalTag/GuardDutyManaged}",
                    "aws:PrincipalArn": "arn:aws:iam::123456789012:role/
org-admins/iam-admin"
                },
                "ForAnyValue:StringEquals": {
                    "aws:TagKeys": [
                        "GuardDutyManaged"
                    ]
                }
            }
        },
        {
            "Sid": "DenyModifyTagsIfPrinTagNotExists",
            "Effect": "Deny",
            "Action": [
                "ecs:TagResource",
                "ecs:UntagResource"
            ],
            "Resource": [
                "*"
            ],
            "Condition": {
                "StringNotEquals": {
                    "aws:PrincipalArn": "arn:aws:iam::123456789012:role/
org-admins/iam-admin"
                },
                "Null": {
                    "aws:PrincipalTag/GuardDutyManaged": true
                }
            }
        }
    ]
}
```

> **Note**
>
> When using inclusion tags for your Amazon ECS clusters, you don't need to enable **Automated agent configuration** explicitly.

4. When you want GuardDuty to monitor tasks that are part of a service, it requires a new service deployment after you enable Runtime Monitoring. If the last deployment for a specific ECS service was started before you enabled Runtime Monitoring, you can either restart the service, or update the service by using `forceNewDeployment`.

   For steps to update the service, see the following resources:

   - [Updating an Amazon ECS service using the console]() in the *Amazon Elastic Container Service Developer Guide*.
   - [UpdateService]() in the *Amazon Elastic Container Service API Reference*.
   - [update-service]() in the *AWS CLI Command Reference*.

Configuring GuardDuty agent for a standalone account

1. Sign in to the AWS Management Console and open the GuardDuty console at https://console.aws.amazon.com/guardduty/.
2. In the navigation pane, choose **Runtime Monitoring**.
3. Under the **Configuration** tab:

   a. **To manage Automated agent configuration for all Amazon ECS clusters (account level)**

      Choose **Enable** in the **Automated agent configuration** section for **AWS Fargate (ECS only)**. When a new Fargate Amazon ECS task launches, GuardDuty will manage the deployment of the security agent.

      - Choose **Save**.

   b. **To manage Automated agent configuration by excluding some of the Amazon ECS clusters (cluster level)**

      i. Add a tag to the Amazon ECS cluster for which you want to exclude all of the tasks. The key-value pair must be `GuardDutyManaged`-`false`.

ii. Prevent modification of these tags, except by trusted entities. The policy provided in Prevent tags from being modified except by authorized principles in the *AWS Organizations User Guide* has been modified to be applicable here.

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "DenyModifyTagsIfResAuthzTagAndPrinTagDontMatch",
            "Effect": "Deny",
            "Action": [
                "ecs:TagResource",
                "ecs:UntagResource"
            ],
            "Resource": [
                "*"
            ],
            "Condition": {
                "StringNotEquals": {
                    "ecs:ResourceTag/GuardDutyManaged":
 "${aws:PrincipalTag/GuardDutyManaged}",
                    "aws:PrincipalArn":
 "arn:aws:iam::123456789012:role/org-admins/iam-admin"
                },
                "Null": {
                    "ecs:ResourceTag/GuardDutyManaged": false
                }
            }
        },
        {
            "Sid": "DenyModifyResAuthzTagIfPrinTagDontMatch",
            "Effect": "Deny",
            "Action": [
                "ecs:TagResource",
                "ecs:UntagResource"
            ],
            "Resource": [
                "*"
            ],
            "Condition": {
```

```
                "StringNotEquals": {
                    "aws:RequestTag/GuardDutyManaged":
 "${aws:PrincipalTag/GuardDutyManaged}",
                    "aws:PrincipalArn":
 "arn:aws:iam::123456789012:role/org-admins/iam-admin"
                },
                "ForAnyValue:StringEquals": {
                    "aws:TagKeys": [
                        "GuardDutyManaged"
                    ]
                }
            }
        },
        {
            "Sid": "DenyModifyTagsIfPrinTagNotExists",
            "Effect": "Deny",
            "Action": [
                "ecs:TagResource",
                "ecs:UntagResource"
            ],
            "Resource": [
                "*"
            ],
            "Condition": {
                "StringNotEquals": {
                    "aws:PrincipalArn":
 "arn:aws:iam::123456789012:role/org-admins/iam-admin"
                },
                "Null": {
                    "aws:PrincipalTag/GuardDutyManaged": true
                }
            }
        }
    ]
}
```

iii. Under the **Configuration** tab, choose **Enable** in the **Automated agent configuration** section.

> **Note**
>
> Always add the exclusion tag to your Amazon ECS cluster before enabling GuardDuty agent auto-management for your account; otherwise, the

security agent will be deployed in all the tasks that are launched within the corresponding Amazon ECS cluster.

For the Amazon ECS clusters that have not been excluded, GuardDuty will manage the deployment of the security agent in the sidecar container.

iv. Choose **Save**.

c. **To manage Automated agent configuration by including some of the Amazon ECS clusters (cluster level)**

i. Add a tag to an Amazon ECS cluster for which you want to include all of the tasks. The key-value pair must be `GuardDutyManaged`-`true`.

ii. Prevent modification of these tags, except by trusted entities. The policy provided in Prevent tags from being modified except by authorized principles in the *AWS Organizations User Guide* has been modified to be applicable here.

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "DenyModifyTagsIfResAuthzTagAndPrinTagDontMatch",
            "Effect": "Deny",
            "Action": [
                "ecs:TagResource",
                "ecs:UntagResource"
            ],
            "Resource": [
                "*"
            ],
            "Condition": {
                "StringNotEquals": {
                    "ecs:ResourceTag/GuardDutyManaged":
 "${aws:PrincipalTag/GuardDutyManaged}",
                    "aws:PrincipalArn":
 "arn:aws:iam::123456789012:role/org-admins/iam-admin"
                },
                "Null": {
```

```
                    "ecs:ResourceTag/GuardDutyManaged": false
                }
            }
        },
        {
            "Sid": "DenyModifyResAuthzTagIfPrinTagDontMatch",
            "Effect": "Deny",
            "Action": [
                "ecs:TagResource",
                "ecs:UntagResource"
            ],
            "Resource": [
                "*"
            ],
            "Condition": {
                "StringNotEquals": {
                    "aws:RequestTag/GuardDutyManaged":
 "${aws:PrincipalTag/GuardDutyManaged}",
                    "aws:PrincipalArn":
 "arn:aws:iam::123456789012:role/org-admins/iam-admin"
                },
                "ForAnyValue:StringEquals": {
                    "aws:TagKeys": [
                        "GuardDutyManaged"
                    ]
                }
            }
        },
        {
            "Sid": "DenyModifyTagsIfPrinTagNotExists",
            "Effect": "Deny",
            "Action": [
                "ecs:TagResource",
                "ecs:UntagResource"
            ],
            "Resource": [
                "*"
            ],
            "Condition": {
                "StringNotEquals": {
                    "aws:PrincipalArn":
 "arn:aws:iam::123456789012:role/org-admins/iam-admin"
                },
                "Null": {
```

```
                            "aws:PrincipalTag/GuardDutyManaged": true
                        }
                    }
                }
            ]
}
```

4. When you want GuardDuty to monitor tasks that are part of a service, it requires a new service deployment after you enable Runtime Monitoring. If the last deployment for a specific ECS service was started before you enabled Runtime Monitoring, you can either restart the service, or update the service by using `forceNewDeployment`.

   For steps to update the service, see the following resources:

   - Updating an Amazon ECS service using the console in the *Amazon Elastic Container Service Developer Guide*.
   - UpdateService in the *Amazon Elastic Container Service API Reference*.
   - update-service in the *AWS CLI Command Reference*.

Managing security agent automatically for Amazon EKS resources

Runtime Monitoring supports enabling the security agent through GuardDuty automated configuration and manually. This section provides the steps to enable automated agent configuration for Amazon EKS clusters.

Before proceeding, make sure that you have followed the Prerequisites for Amazon EKS cluster support.

Based on your preferred approach on how to Manage security agent through GuardDuty, choose the steps in the following sections accordingly.

Configuring Automated agent for multi-account environments

In a multiple-account environments, only the delegated GuardDuty administrator account can enable or disable Automated agent configuration for the member accounts, and manage Automated agent for the EKS clusters belonging to the member accounts in their organization. The GuardDuty member accounts can't modify this configuration from their accounts. The delegated GuardDuty administrator account account manages their member accounts using AWS Organizations. For more information about multi-account environments, see Managing multiple accounts.

Configuring Automated agent configuration for delegated GuardDuty administrator account

<table>
<tr><th>Preferred approach to manage GuardDuty security agent</th><th>Steps</th></tr>
<tr><td>Manage security agent through GuardDuty

(Monitor all EKS clusters)</td><td>If you chose Enable for all accounts in the Runtime Monitoring section, then you have the following options:
Choose Enable for all accounts in the Automated agent configuration section. GuardDuty will deploy and manage the security agent for all the EKS clusters that belong to the delegated GuardDuty administrator account account and also for all the EKS clusters that belong to all the existing and potentially new member accounts in the organization.
Choose Configure accounts manually.
If you chose Configure accounts manually in the Runtime Monitoring section, then do the following:
Choose Configure accounts manually in the Automated agent configuration section.
Choose Enable in the delegated GuardDuty administrator account (this account) section.
Choose Save.</td></tr>
<tr><td>Monitor all EKS clusters but exclude some of them (using exclusion tags)</td><td>From the following procedures, choose one of the scenarios that apply to you.

To exclude an EKS cluster from monitoring when the GuardDuty security agent has not been deployed on this cluster
Add a tag to this EKS cluster with the key as <code>GuardDuty Managed</code> and its value as <code>false</code>.</td></tr>
</table>

Preferred approach to manage GuardDuty security agent	Steps
	For more information about tagging your Amazon EKS cluster, see Working with tags using the console in the Amazon EKS User Guide. 2. To prevent modification of tags, except by the trusted entities, use the policy provided in Prevent tags from being modified except by authorized principals in the *AWS Organizations User Guide*. In this policy, replace the following details: • Replace *ec2:CreateTags* with `eks:TagResource` . • Replace *ec2:DeleteTags* with `eks:UntagResource` . • Replace *access-project* with `GuardDutyManaged` • Replace *123456789012* with the AWS account ID of the trusted entity. When you have more than one trusted entities, use the following example to add multiple `PrincipalArn` : `"aws:PrincipalArn":["arn:aws:iam::123456789012:role/org-admins/iam-admin", "arn:aws:iam::123456789012:role/org-admins/iam-admin", "arn:aws:iam::123456789012:role/org-admins/iam-admin"]` 3. Open the GuardDuty console at https://console.aws.amazon.com/guardduty/. 4. In the navigation pane, choose **Runtime Monitoring**. **Note** Always add the exclusion tag to your EKS clusters before enabling GuardDuty agent auto-management for your account; otherwise, the GuardDuty security

Preferred approach to manage GuardDuty security agent	Steps
	agent will be deployed on all the EKS clusters in your account. 5. Under the **Configuration** tab, choose **Enable** in the **GuardDuty agent management** section. For the EKS clusters that have not been excluded from monitoring, GuardDuty will manage the deployment of and updates to the GuardDuty security agent. 6. Choose **Save**. **To exclude an EKS cluster from monitoring when the GuardDuty security agent has been deployed on this cluster** 1. Add a tag to this EKS cluster with the key as `GuardDuty Managed` and its value as `false`. For more information about tagging your Amazon EKS cluster, see [Working with tags using the console] in the Amazon EKS User Guide. 2. To prevent modification of tags, except by the trusted entities, use the policy provided in [Prevent tags from being modified except by authorized principals] in the *AWS Organizations User Guide*. In this policy, replace the following details: • Replace *`ec2:CreateTags`* with `eks:TagResource` . • Replace *`ec2:DeleteTags`* with `eks:UntagResource` . • Replace *`access-project`* with `GuardDutyManaged` • Replace *`123456789012`* with the AWS account ID of the trusted entity. When you have more than one trusted entities, use the following example to add multiple `PrincipalArn` :

Preferred approach to manage GuardDuty security agent	Steps
	`"aws:PrincipalArn":["arn:aws:iam::123456789012:role/org-admins/iam-admin", "arn:aws:iam::123456789012:role/org-admins/iam-admin", "arn:aws:iam::123456789012:role/org-admins/iam-admin"]` 3. If you had automated agent enabled for this EKS cluster, then after this step, GuardDuty will not update the security agent for this cluster. However, the security agent will remain deployed and GuardDuty will keep on receiving the runtime events from this EKS cluster. This may impact your usage statistics. To stop receiving the runtime events from this cluster, you must remove the deployed security agent from this EKS cluster. For more information about removing the deployed security agent, see Disabling, uninstalling, and cleaning up resources in Runtime Monitoring 4. If you were managing the GuardDuty security agent for this EKS cluster manually, then see Disabling, uninstalling, and cleaning up resources in Runtime Monitoring.

Preferred approach to manage GuardDuty security agent	Steps
Monitor selective EKS clusters using inclusion tags	Regardless of how you chose to enable Runtime Monitoring, the following steps will help you monitor selective EKS clusters in your account: 1. Make sure to choose **Disable** for **delegated GuardDuty administrator account (this account)** in the Automated agent configuration section. Keep the Runtime Monitoring configura tion the same as configured in the previous step. 2. Choose **Save**. 3. Add a tag to your EKS cluster with the key as `GuardDuty Managed` and its value as `true`. For more information about tagging your Amazon EKS cluster, see [Working with tags using the console](#) in the Amazon EKS User Guide. GuardDuty will manage the deployment of and updates to the security agent for the selective EKS clusters that you want to monitor. 4. To prevent modification of tags, except by the trusted entities, use the policy provided in Prevent tags from being modified except by authorized principals in the *AWS Organizations User Guide*. In this policy, replace the following details: • Replace *`ec2:CreateTags`* with `eks:TagResource` . • Replace *`ec2:DeleteTags`* with `eks:UntagResource` . • Replace *`access-project`* with `GuardDutyManaged` • Replace *`123456789012`* with the AWS account ID of the trusted entity. When you have more than one trusted entities, use the following example to add multiple `PrincipalArn` :

Preferred approach to manage GuardDuty security agent	Steps
	`"aws:PrincipalArn":["arn:aws:iam::123456789012:role/org-admins/iam-admin", "arn:aws:iam::123456789012:role/org-admins/iam-admin", "arn:aws:iam::123456789012:role/org-admins/iam-admin"]`
Manage the GuardDuty security agent manually	Regardless of how you chose to enable Runtime Monitoring, you can manage the security agent manually for your EKS clusters. 1. Make sure to choose **Disable** for **delegated GuardDuty administrator account (this account)** in the Automated agent configuration section. Keep the Runtime Monitoring configuration the same as configured in the previous step. 2. Choose **Save**. 3. To manage the security agent, see Managing security agent manually for Amazon EKS cluster.

Auto-enable Automated agent for all member accounts

> **Note**
>
> It may take up to 24 hours to update the configuration for the member accounts.

Preferred approach to manage GuardDuty security agent	Steps
Manage security agent through GuardDuty (Monitor all EKS clusters)	This topic is to enable Runtime Monitoring for all member accounts and therefore, the following steps assume that you must have chosen **Enable for all accounts** in the Runtime Monitoring section.

Preferred approach to manage GuardDuty security agent	Steps
	1. Choose **Enable for all accounts** in the Automated agent configuration section. GuardDuty will deploy and manage the security agent for all the EKS clusters that belong to the delegated GuardDuty administrator account account and also for all the EKS clusters that belong to all the existing and potentially new member accounts in the organization. 2. Choose **Save**.

Preferred approach to manage GuardDuty security agent	Steps
Monitor all EKS clusters but exclude some of them (using exclusion tags)	From the following procedures, choose one of the scenarios that apply to you. **To exclude an EKS cluster from monitoring when the GuardDuty security agent has not been deployed on this cluster** 1. Add a tag to this EKS cluster with the key as `GuardDutyManaged` and its value as `false`. For more information about tagging your Amazon EKS cluster, see [Working with tags using the console]() in the Amazon EKS User Guide. 2. To prevent modification of tags, except by the trusted entities, use the policy provided in [Prevent tags from being modified except by authorized principals]() in the *AWS Organizations User Guide*. In this policy, replace the following details: • Replace *`ec2:CreateTags`* with `eks:TagResource` . • Replace *`ec2:DeleteTags`* with `eks:UntagResource` . • Replace *`access-project`* with `GuardDutyManaged` • Replace *`123456789012`* with the AWS account ID of the trusted entity. When you have more than one trusted entities, use the following example to add multiple `PrincipalArn` : `"aws:PrincipalArn":["arn:aws:iam::123456789012:role/org-admins/iam-admin", "arn:aws:iam::123456789012:role/org-admins/iam-admin", "arn:aws:iam::123456789012:role/org-admins/iam-admin"]` 3. Open the GuardDuty console at https://console.aws.amazon.com/guardduty/. 4. In the navigation pane, choose **Runtime Monitoring**.

Preferred approach to manage GuardDuty security agent	Steps
	Note Always add the exclusion tag to your EKS clusters before enabling Automated agent for your account; otherwise, the GuardDuty security agent will be deployed on all the EKS clusters in your account. 5. Under the **Configuration** tab, choose **Edit** in the **Runtime Monitoring configuration** section. 6. Choose **Enable for all accounts** in the Automated agent configuration section. For the EKS clusters that have not been excluded from monitoring, GuardDuty will manage the deployment of and updates to the GuardDuty security agent. 7. Choose **Save**. **To exclude an EKS cluster from monitoring when the GuardDuty security agent has been deployed on this cluster** 1. Add a tag to this EKS cluster with the key as `GuardDuty Managed` and its value as `false`. For more information about tagging your Amazon EKS cluster, see [Working with tags using the console]() in the Amazon EKS User Guide. 2. If you had Automated agent configuration enabled for this EKS cluster, then after this step, GuardDuty will not update the security agent for this cluster. However, the security agent will remain deployed and GuardDuty will keep on receiving the runtime events from this EKS cluster. This may impact your usage statistics.

Preferred approach to manage GuardDuty security agent	Steps
	To stop receiving the runtime events from this cluster, you must remove the deployed security agent from this EKS cluster. For more information about removing the deployed security agent, see Disabling, uninstalling, and cleaning up resources in Runtime Monitoring 3. To prevent modification of tags, except by the trusted entities, use the policy provided in Prevent tags from being modified except by authorized principals in the *AWS Organizations User Guide*. In this policy, replace the following details: • Replace *ec2:CreateTags* with `eks:TagResource`. • Replace *ec2:DeleteTags* with `eks:UntagResource`. • Replace *access-project* with `GuardDutyManaged` • Replace *123456789012* with the AWS account ID of the trusted entity. When you have more than one trusted entities, use the following example to add multiple `PrincipalArn`: `"aws:PrincipalArn":["arn:aws:iam::123456789012:role/org-admins/iam-admin", "arn:aws:iam::123456789012:role/org-admins/iam-admin", "arn:aws:iam::123456789012:role/org-admins/iam-admin"]` 4. If you were managing the GuardDuty security agent for this EKS cluster manually, then see Disabling, uninstalling, and cleaning up resources in Runtime Monitoring.

Preferred approach to manage GuardDuty security agent	Steps
Monitor selective EKS clusters using inclusion tags	Regardless of how you chose to enable Runtime Monitoring, the following steps will help you monitor selective EKS clusters for all member accounts in your organization: 1. Do not enable any configuration in the Automated agent configuration section. Keep the Runtime Monitoring configura tion the same as configured in the previous step. 2. Choose **Save**. 3. Add a tag to your EKS cluster with the key as `GuardDuty Managed` and its value as `true`. For more information about tagging your Amazon EKS cluster, see [Working with tags using the console] in the Amazon EKS User Guide. GuardDuty will manage the deployment of and updates to the security agent for the selective EKS clusters that you want to monitor. 4. To prevent modification of tags, except by the trusted entities, use the policy provided in [Prevent tags from being modified except by authorized principals] in the *AWS Organizations User Guide*. In this policy, replace the following details: • Replace *`ec2:CreateTags`* with `eks:TagResource` . • Replace *`ec2:DeleteTags`* with `eks:UntagResource` . • Replace *`access-project`* with `GuardDutyManaged` • Replace *`123456789012`* with the AWS account ID of the trusted entity. When you have more than one trusted entities, use the following example to add multiple `PrincipalArn` :

Preferred approach to manage GuardDuty security agent	Steps
	`"aws:PrincipalArn":["arn:aws:iam::123456789012:role/org-admins/iam-admin", "arn:aws:iam::123456789012:role/org-admins/iam-admin", "arn:aws:iam::123456789012:role/org-admins/iam-admin"]`
Manage the GuardDuty security agent manually	Regardless of how you chose to enable Runtime Monitoring, you can manage the security agent manually for your EKS clusters. 1. Do not enable any configuration in the Automated agent configuration section. Keep the Runtime Monitoring configuration the same as configured in the previous step. 2. Choose **Save**. 3. To manage the security agent, see Managing security agent manually for Amazon EKS cluster.

Enabling automated agent for all existing active member accounts

> **Note**
>
> It may take up to 24 hours to update the configuration for the member accounts.

To manage GuardDuty security agent for existing active member accounts in your organization

- For GuardDuty to receive the runtime events from the EKS clusters that belong to the existing active member accounts in the organization, you must choose a preferred approach to manage the GuardDuty security agent for these EKS clusters. For more information about each of these approaches, see Approaches to manage GuardDuty security agent in Amazon EKS clusters.

Preferred approach to manage GuardDuty security agent	Steps
Manage security agent through GuardDuty (Monitor all EKS clusters)	**To monitor all EKS clusters for all existing active member accounts** 1. On the Runtime Monitoring page, under the **Configuration** tab, you can view the current status of Automated agent configuration. 2. Within the **Automated agent configuration** pane, under the **Active member accounts** section, choose **Actions**. 3. From **Actions**, choose **Enable for all existing active member accounts**. 4. Choose **Confirm**.

<table>
<tr><th>Preferred approach to manage GuardDuty security agent</th><th>Steps</th></tr>
<tr><td>Monitor all EKS clusters but exclude some of them (using exclusion tag)</td><td>

From the following procedures, choose one of the scenarios that apply to you.

To exclude an EKS cluster from monitoring when the GuardDuty security agent has not been deployed on this cluster

1. Add a tag to this EKS cluster with the key as `GuardDutyManaged` and its value as `false`.

   For more information about tagging your Amazon EKS cluster, see Working with tags using the console in the Amazon EKS User Guide.
2. To prevent modification of tags, except by the trusted entities, use the policy provided in Prevent tags from being modified except by authorized principals in the *AWS Organizations User Guide*. In this policy, replace the following details:

   - Replace *`ec2:CreateTags`* with `eks:TagResource`.
   - Replace *`ec2:DeleteTags`* with `eks:UntagResource`.
   - Replace *`access-project`* with `GuardDutyManaged`
   - Replace *`123456789012`* with the AWS account ID of the trusted entity.

     When you have more than one trusted entities, use the following example to add multiple `PrincipalArn`:

```
"aws:PrincipalArn":["arn:aws:iam::123456789012:role/org-admins/iam-admin", "arn:aws:iam::123456789012:
```
</td></tr>
</table>

Preferred approach to manage GuardDuty security agent	Steps
	```role/org-admins/iam-admin", "arn:aws:iam::123456789012:role/org-admins/iam-admin"]``` 3. Open the GuardDuty console at https://console.aws.amazon.com/guardduty/. 4. In the navigation pane, choose **Runtime Monitoring**. **Note** Always add the exclusion tag to your EKS clusters before enabling Automated agent configuration for your account; otherwise, the GuardDuty security agent will be deployed on all the EKS clusters in your account. 5. Under the **Configuration** tab, in the **Automated agent configuration** pane, under **Active member accounts**, choose **Actions**. 6. From **Actions**, choose **Enable for all active member accounts**. 7. Choose **Confirm**. **To exclude an EKS cluster from monitoring after the GuardDuty security agent has already been deployed on this cluster** 1. Add a tag to this EKS cluster with the key as `GuardDutyManaged` and its value as `false`. For more information about tagging your Amazon EKS cluster, see [Working with tags using the console](#) in the Amazon EKS User Guide.

Preferred approach to manage GuardDuty security agent	Steps
	After this step, GuardDuty will not update the security agent for this cluster. However, the security agent will remain deployed and GuardDuty will keep on receiving the runtime events from this EKS cluster. This may impact your usage statistics. 2. To prevent modification of tags, except by the trusted entities, use the policy provided in [Prevent tags from being modified except by authorized principals](#) in the *AWS Organizations User Guide*. In this policy, replace the following details: • Replace *ec2:CreateTags* with `eks:TagResource`. • Replace *ec2:DeleteTags* with `eks:UntagResource`. • Replace *access-project* with `GuardDutyManaged` • Replace *123456789012* with the AWS account ID of the trusted entity. When you have more than one trusted entities, use the following example to add multiple `PrincipalArn`: `"aws:PrincipalArn":["arn:aws:iam::123456789012:role/org-admins/iam-admin", "arn:aws:iam::123456789012:role/org-admins/iam-admin", "arn:aws:iam::123456789012:role/org-admins/iam-admin"]` 3. Regardless of how you manage the security agent (through GuardDuty or manually), to stop receiving the runtime events from this cluster, you must

Preferred approach to manage GuardDuty security agent	Steps
	remove the deployed security agent from this EKS cluster. For more information about removing the deployed security agent, see Disabling, uninstalling, and cleaning up resources in Runtime Monitoring.

Preferred approach to manage GuardDuty security agent	Steps
Monitor selective EKS clusters using inclusion tags	1. On the Accounts page, after you enable Runtime Monitoring, do not enable **Runtime Monitoring - Automated agent configuration**. 2. Add a tag to the EKS cluster that belongs to the selected account that you want to monitor. The key-value pair of the tag must be `GuardDuty Managed` -`true`. For more information about tagging your Amazon EKS cluster, see [Working with tags using the console]() in the Amazon EKS User Guide. GuardDuty will manage the deployment of and updates to the security agent for the selective EKS clusters that you want to monitor. 3. To prevent modification of tags, except by the trusted entities, use the policy provided in [Prevent tags from being modified except by authorized principals]() in the *AWS Organizations User Guide*. In this policy, replace the following details: • Replace *`ec2:CreateTags`* with `eks:TagResource` . • Replace *`ec2:DeleteTags`* with `eks:UntagResource` . • Replace *`access-project`* with `GuardDuty Managed` • Replace *`123456789012`* with the AWS account ID of the trusted entity. When you have more than one trusted entities, use the following example to add multiple `PrincipalArn` :

Preferred approach to manage GuardDuty security agent	Steps
	`"aws:PrincipalArn":["arn:aws:iam::123456789012:role/org-admins/iam-admin", "arn:aws:iam::123456789012:role/org-admins/iam-admin", "arn:aws:iam::123456789012:role/org-admins/iam-admin"]`
Manage the GuardDuty security agent manually	1. Make sure you don't choose **Enable** in the **Automated agent configuration** section. Keep Runtime Monitoring enabled. 2. Choose **Save**. 3. To manage the security agent, see Managing security agent manually for Amazon EKS cluster.

Auto-enable automated agent configuration for new members

Preferred approach to manage GuardDuty security agent	Steps
Manage security agent through GuardDuty (Monitor all EKS clusters)	1. On the Runtime Monitoring page, choose **Edit** to update the existing configuration. 2. In the Automated agent configuration section, select **Automatically enable for new member accounts**. 3. Choose **Save**.
Monitor all EKS clusters but exclude some of them (using exclusion tags)	From the following procedures, choose one of the scenarios that apply to you.

Preferred approach to manage GuardDuty security agent	Steps
	To exclude an EKS cluster from monitoring when the GuardDuty security agent has not been deployed on this cluster 1. Add a tag to this EKS cluster with the key as `GuardDutyManaged` and its value as `false`. For more information about tagging your Amazon EKS cluster, see Working with tags using the console in the Amazon EKS User Guide. 2. To prevent modification of tags, except by the trusted entities, use the policy provided in Prevent tags from being modified except by authorized principals in the *AWS Organizations User Guide*. In this policy, replace the following details: • Replace *`ec2:CreateTags`* with `eks:TagResource`. • Replace *`ec2:DeleteTags`* with `eks:UntagResource`. • Replace *`access-project`* with `GuardDutyManaged` • Replace *`123456789012`* with the AWS account ID of the trusted entity. When you have more than one trusted entities, use the following example to add multiple `PrincipalArn`: `"aws:PrincipalArn":["arn:aws:iam::123456789012:role/org-admins/iam-admin", "arn:aws:iam::123456789012:role/org-admins/iam-admin", "arn:aws:iam::123456789012:role/org-admins/iam-admin"]`

Preferred approach to manage GuardDuty security agent	Steps
	3. Open the GuardDuty console at https://console.aws.amazon.com/guardduty/. 4. In the navigation pane, choose **Runtime Monitoring**. **Note** Always add the exclusion tag to your EKS clusters before enabling Automated agent configuration for your account; otherwise, the GuardDuty security agent will be deployed on all the EKS clusters in your account. 5. Under the **Configuration** tab, select **Automatically enable for new member accounts** in the **GuardDuty agent management** section. For the EKS clusters that have not been excluded from monitoring, GuardDuty will manage the deployment of and updates to the GuardDuty security agent. 6. Choose **Save**. **To exclude an EKS cluster from monitoring when the GuardDuty security agent has been deployed on this cluster** 1. Regardless of whether you manage the GuardDuty security agent through GuardDuty or manually, add a tag to this EKS cluster with the key as `GuardDutyManaged` and its value as `false`. For more information about tagging your Amazon EKS cluster, see [Working with tags using the console](#) in the Amazon EKS User Guide.

Preferred approach to manage GuardDuty security agent	Steps
	If you had Automated agent enabled for this EKS cluster, then after this step, GuardDuty will not update the security agent for this cluster. However, the security agent will remain deployed and GuardDuty will keep on receiving the runtime events from this EKS cluster. This may impact your usage statistics. To stop receiving the runtime events from this cluster, you must remove the deployed security agent from this EKS cluster. For more information about removing the deployed security agent, see [Disabling, uninstalling, and cleaning up resources in Runtime Monitoring]() 2. To prevent modification of tags, except by the trusted entities, use the policy provided in [Prevent tags from being modified except by authorized principals]() in the *AWS Organizations User Guide*. In this policy, replace the following details: • Replace *`ec2:CreateTags`* with `eks:TagResource` . • Replace *`ec2:DeleteTags`* with `eks:UntagResource` . • Replace *`access-project`* with `GuardDutyManaged` • Replace *`123456789012`* with the AWS account ID of the trusted entity. When you have more than one trusted entities, use the following example to add multiple `PrincipalArn` : `"aws:PrincipalArn":["arn:aws:iam::123456789012:role/org-admins/iam-admin", "arn:aws:iam::123456789012:role/org-`

Preferred approach to manage GuardDuty security agent	Steps
	`admins/iam-admin", "arn:aws:iam::123456789012:role/org-admins/iam-admin"]` 3. If you were managing the GuardDuty security agent for this EKS cluster manually, then see Disabling, uninstalling, and cleaning up resources in Runtime Monitoring.

<table>
<tr><th>Preferred approach to manage GuardDuty security agent</th><th>Steps</th></tr>
<tr><td>Monitor selective EKS clusters using inclusion tags</td><td>Regardless of how you chose to enable Runtime Monitoring, the following steps will help you monitor selective EKS clusters for the new member accounts in your organization.

1. Make sure to clear Automatically enable for new member accounts in the Automated agent configuration section. Keep the Runtime Monitoring configuration the same as configured in the previous step.
2. Choose Save.
3. Add a tag to your EKS cluster with the key as <code>GuardDutyManaged</code> and its value as <code>true</code>.

For more information about tagging your Amazon EKS cluster, see Working with tags using the console in the Amazon EKS User Guide.

GuardDuty will manage the deployment of and updates to the security agent for the selective EKS clusters that you want to monitor.
4. To prevent modification of tags, except by the trusted entities, use the policy provided in Prevent tags from being modified except by authorized principals in the AWS Organizations User Guide. In this policy, replace the following details:

• Replace <code>ec2:CreateTags</code> with <code>eks:TagResource</code> .
• Replace <code>ec2:DeleteTags</code> with <code>eks:UntagResource</code> .
• Replace <code>access-project</code> with <code>GuardDutyManaged</code></td></tr>
</table>

Preferred approach to manage GuardDuty security agent	Steps
	• Replace *123456789012* with the AWS account ID of the trusted entity. When you have more than one trusted entities, use the following example to add multiple `Principal Arn` : `"aws:PrincipalArn":["arn:aws:iam::123456789012:role/org-admins/iam-admin", "arn:aws:iam::123456789012:role/org-admins/iam-admin", "arn:aws:iam::123456789012:role/org-admins/iam-admin"]`
Manage the GuardDuty security agent manually	Regardless of how you chose to enable Runtime Monitoring, you can manage the security agent manually for your EKS clusters. 1. Make sure clear the checkbox **Automatically enable for new member accounts** in the Automated agent configuration section. Keep the Runtime Monitoring configuration the same as configured in the previous step. 2. Choose **Save**. 3. To manage the security agent, see Managing security agent manually for Amazon EKS cluster.

Configuring Automated agent for active member accounts selectively

Preferred approach to manage GuardDuty security agent	Steps
Manage security agent through GuardDuty	1. On the Accounts page, select the accounts for which you want to enable Automated agent configuration. You can select

Preferred approach to manage GuardDuty security agent	Steps
(Monitor all EKS clusters)	more than one account at a time. Make sure that the accounts you select in this step already have EKS Runtime Monitoring enabled. 2. From **Edit Protection plans** choose the appropriate option to enable **Runtime Monitoring - Automated agent configuration**. 3. Choose **Confirm**.

Preferred approach to manage GuardDuty security agent	Steps
Monitor all EKS clusters but exclude some of them (using exclusion tags)	From the following procedures, choose one of the scenarios that apply to you. **To exclude an EKS cluster from monitoring when the GuardDuty security agent has not been deployed on this cluster** 1. Add a tag to this EKS cluster with the key as `GuardDuty Managed` and its value as `false`. For more information about tagging your Amazon EKS cluster, see Working with tags using the console in the Amazon EKS User Guide. 2. To prevent modification of tags, except by the trusted entities, use the policy provided in Prevent tags from being modified except by authorized principals in the *AWS Organizations User Guide*. In this policy, replace the following details: • Replace *`ec2:CreateTags`* with `eks:TagResource` . • Replace *`ec2:DeleteTags`* with `eks:UntagResource` . • Replace *`access-project`* with `GuardDutyManaged` • Replace *`123456789012`* with the AWS account ID of the trusted entity. When you have more than one trusted entities, use the following example to add multiple `PrincipalArn` : `"aws:PrincipalArn":["arn:aws:iam::123456789012:role/org-admins/iam-admin", "arn:aws:iam::123456789012:role/org-admins/iam-admin", "arn:aws:iam::123456789012:role/org-admins/iam-admin"]` 3. Open the GuardDuty console at https://console.aws.amazon.com/guardduty/.

Preferred approach to manage GuardDuty security agent	Steps
	Note Always add the exclusion tag to your EKS clusters before enabling Automated agent configuration for your account; otherwise, the GuardDuty security agent will be deployed on all the EKS clusters in your account. 4. On the Accounts page, select the account for which you want to enable **Manage agent automatically**. You can select more than one account at a time. 5. From **Edit protection plans**, choose the appropriate option to enable **Runtime Monitoring-Automated agent configuration** for the selected account. For the EKS clusters that have not been excluded from monitoring, GuardDuty will manage the deployment of and updates to the GuardDuty security agent. 6. Choose **Save**. **To exclude an EKS cluster from monitoring when the GuardDuty security agent has been deployed on this cluster** 1. Add a tag to this EKS cluster with the key as `GuardDuty Managed` and its value as `false`. For more information about tagging your Amazon EKS cluster, see [Working with tags using the console](#) in the Amazon EKS User Guide. If you had previously Automated agent configuration enabled for this EKS cluster, then after this step, GuardDuty will not update the security agent for this cluster. However, the security agent will remain deployed and GuardDuty will keep on

Preferred approach to manage GuardDuty security agent	Steps
	receiving the runtime events from this EKS cluster. This may impact your usage statistics. To stop receiving the runtime events from this cluster, you must remove the deployed security agent from this EKS cluster. For more information about removing the deployed security agent, see Disabling, uninstalling, and cleaning up resources in Runtime Monitoring 2. To prevent modification of tags, except by the trusted entities, use the policy provided in Prevent tags from being modified except by authorized principals in the *AWS Organizations User Guide*. In this policy, replace the following details: • Replace *ec2:CreateTags* with `eks:TagResource` . • Replace *ec2:DeleteTags* with `eks:UntagResource` . • Replace *access-project* with `GuardDutyManaged` • Replace *123456789012* with the AWS account ID of the trusted entity. When you have more than one trusted entities, use the following example to add multiple `PrincipalArn` : `"aws:PrincipalArn":["arn:aws:iam::123456789012:role/org-admins/iam-admin", "arn:aws:iam::123456789012:role/org-admins/iam-admin", "arn:aws:iam::123456789012:role/org-admins/iam-admin"]` 3. If you were managing the GuardDuty security agent for this EKS cluster manually, you must remove it. For more information, see Disabling, uninstalling, and cleaning up resources in Runtime Monitoring.

Preferred approach to manage GuardDuty security agent	Steps
Monitor selective EKS clusters using inclusion tags	Regardless of how you chose to enable Runtime Monitoring, the following steps will help you monitor selective EKS clusters that belong to the selected accounts: 1. Make sure that you do not enable **Runtime Monitoring- Automated agent configuration** for the selected accounts that have the EKS clusters that you want to monitor. 2. Add a tag to your EKS cluster with the key as `GuardDuty Managed` and its value as `true`. For more information about tagging your Amazon EKS cluster, see [Working with tags using the console] in the Amazon EKS User Guide. After adding the tag, GuardDuty will manage the deploymen t of and updates to the security agent for the selective EKS clusters that you want to monitor. 3. To prevent modification of tags, except by the trusted entities, use the policy provided in [Prevent tags from being modified except by authorized principals] in the *AWS Organizations User Guide*. In this policy, replace the following details: • Replace *`ec2:CreateTags`* with `eks:TagResource` . • Replace *`ec2:DeleteTags`* with `eks:UntagResource` . • Replace *`access-project`* with `GuardDutyManaged` • Replace *`123456789012`* with the AWS account ID of the trusted entity. When you have more than one trusted entities, use the following example to add multiple `PrincipalArn` : `"aws:PrincipalArn":["arn:aws:iam::12345678901 2:role/org-admins/iam-admin", "arn:aws:iam::1234`

Preferred approach to manage GuardDuty security agent	Steps
	`56789012:role/org-admins/iam-admin", "arn:aws:iam::123456789012:role/org-admins/iam-admin"]`
Manage the GuardDuty security agent manually	1. Keep the Runtime Monitoring configuration the same as configured in the previous step. Make sure that you don't enable **Runtime Monitoring- Automated agent configuration** for any of the selected accounts. 2. Choose **Confirm**. 3. To manage the security agent, see Managing security agent manually for Amazon EKS cluster.

Configuring Automated agent for standalone account

A standalone account owns the decision to enable or disable a protection plan in their AWS account in a specific AWS Region.

If your account is associated with a GuardDuty administrator account through AWS Organizations, or by the method of invitation, this section doesn't apply to your account. For more information, see Enabling Runtime Monitoring for multiple-account environments.

After you enable Runtime Monitoring, ensure to install GuardDuty security agent through automated configuration or manual deployment. As a part of completing all the steps listed in the following procedure, make sure to install the security agent.

Based on your preference to monitor all or selective Amazon EKS resources, choose a preferred method and follow the steps in the following table.

1. Sign in to the AWS Management Console and open the GuardDuty console at https://console.aws.amazon.com/guardduty/.
2. In the navigation pane, choose **Runtime Monitoring**.
3. Under the **Configuration** tab, choose **Enable** to enable automated agent configuration for your account.

Preferred approach to deploy GuardDuty security agent	Steps
Manage security agent through GuardDuty (Monitor all EKS clusters)	1. Choose **Enable** in the **Automated agent configura tion** section. GuardDuty will manage the deploymen t of and updates to the security agent for all the existing and potentially new EKS clusters in your account. 2. Choose **Save**.

<table>
<tr><th>Preferred approach to deploy GuardDuty security agent</th><th>Steps</th></tr>
<tr><td>Monitor all EKS clusters but exclude some of them (using exclusion tag)</td><td>
From the following procedures, choose one of the scenarios that is applicable to you.
To exclude an EKS cluster from monitoring when the GuardDuty security agent has not been deployed on this cluster

Add a tag to this EKS cluster with the key as <code>GuardDutyManaged</code> and its value as <code>false</code>.

For more information about tagging your Amazon EKS cluster, see Working with tags using the console in the Amazon EKS User Guide.
To prevent modification of tags, except by the trusted entities, use the policy provided in Prevent tags from being modified except by authorized principals in the AWS Organizations User Guide. In this policy, replace the following details:

Replace <code>ec2:CreateTags</code> with <code>eks:TagResource</code> .
Replace <code>ec2:DeleteTags</code> with <code>eks:UntagResource</code> .
Replace <code>access-project</code> with <code>GuardDutyManaged</code>
Replace <code>123456789012</code> with the AWS account ID of the trusted entity.

When you have more than one trusted entities, use the following example to add multiple <code>PrincipalArn</code> :
<pre>"aws:PrincipalArn":["arn:aws:iam::123456789012:role/org-admins/iam-admin", "arn:aws:iam::123456789012:</pre>

</td></tr>
</table>

Preferred approach to deploy GuardDuty security agent	Steps
	`role/org-admins/iam-admin", "arn:aws:iam::123456789012:role/org-admins/iam-admin"]` 3. Open the GuardDuty console at https://console.aws.amazon.com/guardduty/. 4. In the navigation pane, choose **Runtime Monitoring**. **Note** Always add the exclusion tag to your EKS clusters before enabling GuardDuty agent auto-management for your account; otherwise, the GuardDuty security agent will be deployed on all the EKS clusters in your account. 5. Under the **Configuration** tab, choose **Enable** in the **GuardDuty agent management** section. For the EKS clusters that have not been excluded from monitoring, GuardDuty will manage the deployment of and updates to the GuardDuty security agent. 6. Choose **Save**. **To exclude an EKS cluster from monitoring after the GuardDuty security agent has already been deployed on this cluster** 1. Add a tag to this EKS cluster with the key as `GuardDutyManaged` and its value as `false`.

Preferred approach to deploy GuardDuty security agent	Steps
	For more information about tagging your Amazon EKS cluster, see [Working with tags using the console](#) in the Amazon EKS User Guide. After this step, GuardDuty will not update the security agent for this cluster. However, the security agent will remain deployed and GuardDuty will keep on receiving the runtime events from this EKS cluster. This may impact your usage statistics. 2. To prevent modification of tags, except by the trusted entities, use the policy provided in [Prevent tags from being modified except by authorized principals](#) in the *AWS Organizations User Guide*. In this policy, replace the following details: • Replace *`ec2:CreateTags`* with `eks:TagResource` . • Replace *`ec2:DeleteTags`* with `eks:UntagResource` . • Replace *`access-project`* with `GuardDutyManaged` • Replace *`123456789012`* with the AWS account ID of the trusted entity. When you have more than one trusted entities, use the following example to add multiple `PrincipalArn` : `"aws:PrincipalArn":["arn:aws:iam::123456789012:role/org-admins/iam-admin", "arn:aws:iam::123456789012:role/org-admins/iam-admin", "arn:aws:iam::123456789012:role/org-admins/iam-admin"]`

Preferred approach to deploy GuardDuty security agent	Steps
	3. To stop receiving the runtime events from this cluster, you must remove the deployed security agent from this EKS cluster. For more information about removing the deployed security agent, see Disabling, uninstalling, and cleaning up resources in Runtime Monitoring.

Preferred approach to deploy GuardDuty security agent	Steps
Monitor selective EKS clusters using inclusion tags	1. Make sure to choose **Disable** in the **Automated agent configuration** section. Keep Runtime Monitoring enabled. 2. Choose **Save** 3. Add a tag to this EKS cluster with the key as `GuardDutyManaged` and its value as `true`. For more information about tagging your Amazon EKS cluster, see Working with tags using the console in the Amazon EKS User Guide. GuardDuty will manage the deployment of and updates to the security agent for the selective EKS clusters that you want to monitor. 4. To prevent modification of tags, except by the trusted entities, use the policy provided in Prevent tags from being modified except by authorized principals in the *AWS Organizations User Guide*. In this policy, replace the following details: • Replace *`ec2:CreateTags`* with `eks:TagResource`. • Replace *`ec2:DeleteTags`* with `eks:UntagResource`. • Replace *`access-project`* with `GuardDutyManaged` • Replace *`123456789012`* with the AWS account ID of the trusted entity. When you have more than one trusted entities, use the following example to add multiple `PrincipalArn`:

Preferred approach to deploy GuardDuty security agent	Steps
	`"aws:PrincipalArn":["arn:aws:iam::123456789012:role/org-admins/iam-admin", "arn:aws:iam::123456789012:role/org-admins/iam-admin", "arn:aws:iam::123456789012:role/org-admins/iam-admin"]`
Manage agent manually	1. Make sure to choose **Disable** in the **Automated agent configuration** section. Keep Runtime Monitoring enabled. 2. Choose **Save**. 3. To manage the security agent, see Managing security agent manually for Amazon EKS cluster.

Managing security agent manually for Amazon EKS cluster

This section describes how you can manage your Amazon EKS add-on agent (GuardDuty agent) after you enable Runtime Monitoring (or EKS Runtime Monitoring). To use Runtime Monitoring, you must enable Runtime Monitoring and configure the Amazon EKS add-on, `aws-guardduty-agent`. You require to perform both the steps for GuardDuty to detect potential threats and generate GuardDuty Runtime Monitoring finding types.

For managing the agent manually, you need to create a VPC endpoint as a prerequisite. This helps GuardDuty receive the runtime events. After this, you can install the security agent so that GuardDuty will start receiving the runtime events from the Amazon EKS resources. When GuardDuty releases a new agent version for this resource, you can update the agent version in your account.

Topics

- Prerequisite – Creating an Amazon VPC endpoint
- Installing GuardDuty security agent manually on Amazon EKS resources
- Updating security agent manually for Amazon EKS resources

Prerequisite – Creating an Amazon VPC endpoint

Before you can install the GuardDuty security agent, you must create an Amazon Virtual Private Cloud (Amazon VPC) endpoint. This will help GuardDuty receive the runtime events of your Amazon EKS resources.

> **Note**
>
> There is no additional cost for the usage of the VPC endpoint.

Choose a preferred access method to create an Amazon VPC endpoint.

Console

To create a VPC endpoint

1. Open the Amazon VPC console at https://console.aws.amazon.com/vpc/.
2. In the navigation pane, under **Virtual private cloud**, choose **Endpoints**.
3. Choose **Create Endpoint**.
4. On the **Create endpoint** page, for **Service category**, choose **Other endpoint services**.
5. For **Service name**, enter **`com.amazonaws.us-east-1.guardduty-data`**.

   Make sure to replace *`us-east-1`* with the correct Region. This must be the same Region as the EKS cluster that belongs to your AWS account ID.
6. Choose **Verify service**.
7. After the service name is successfully verified, choose the **VPC** where your cluster resides. Add the following policy to restrict VPC endpoint usage to specified account only. With the organization `Condition` provided below this policy, you can update the following policy to restrict access to your endpoint. To provide VPC endpoint support to specific account IDs in your organization, see Organization condition to restrict access to your endpoint.

   JSON

   ```
   {
    "Version": "2012-10-17",
    "Statement": [
     {
   ```

```
    "Action": "*",
    "Resource": "*",
    "Effect": "Allow",
    "Principal": "*"
   },
   {
    "Condition": {
     "StringNotEquals": {
      "aws:PrincipalAccount": "111122223333"
     }
    },
    "Action": "*",
    "Resource": "*",
    "Effect": "Deny",
    "Principal": "*"
   }
  ]
 }
```

The `aws:PrincipalAccount` account ID must match the account containing the VPC and VPC endpoint. The following list shows how to share the VPC endpoint with other AWS account IDs:

Organization condition to restrict access to your endpoint

- To specify multiple accounts to access the VPC endpoint, replace `"aws:PrincipalAccount": "111122223333"` with the following:

```
"aws:PrincipalAccount": [
          "666666666666",
          "555555555555"
    ]
```

- To allow all the members from an organization to access the VPC endpoint, replace `"aws:PrincipalAccount": "111122223333"` with the following:

```
"aws:PrincipalOrgID": "o-abcdef0123"
```

- To restrict accessing a resource to an organization ID, add your `ResourceOrgID` to the policy.

For more information, see ResourceOrgID.

```
"aws:ResourceOrgID": "o-abcdef0123"
```

8. Under **Additional settings**, choose **Enable DNS name**.
9. Under **Subnets**, choose the subnets in which your cluster resides.
10. Under **Security groups**, choose a security group that has the in-bound port 443 enabled from your VPC (or your EKS cluster). If you don't already have a security group that has an in-bound port 443 enabled, Create a security group.

    If there is an issue while restricting the in-bound permissions to your VPC (or instance), you can the in-bound 443 port from any IP address `(0.0.0.0/0)`. However, GuardDuty recommends using IP addresses that matches the CIDR block for your VPC. For more information, see VPC CIDR blocks in the *Amazon VPC User Guide*.

API/CLI

To create a VPC endpoint

- Invoke CreateVpcEndpoint.
- Use the following values for the parameters:
  - For **Service name**, enter **`com.amazonaws.us-east-1.guardduty-data`**.

    Make sure to replace *`us-east-1`* with the correct Region. This must be the same Region as the EKS cluster that belongs to your AWS account ID.
  - For DNSOptions, enable private DNS option by setting it to `true`.
- For AWS Command Line Interface, see create-vpc-endpoint.

After you have followed the steps, see Validating VPC endpoint configuration to ensure that the VPC endpoint was set up correctly.

Installing GuardDuty security agent manually on Amazon EKS resources

This section describes how you can deploy the GuardDuty security agent for the first time for specific EKS clusters. Before you proceed with this section, make sure you have already set up the prerequisites and enabled Runtime Monitoring for your accounts. The GuardDuty security agent (EKS add-on) will not work if you do not enable Runtime Monitoring.

Choose your preferred access method to deploy the GuardDuty security agent for the first time.

Console

1. Open the Amazon EKS console at https://console.aws.amazon.com/eks/home#/clusters.
2. Choose your **Cluster name**.
3. Choose the **Add-ons** tab.
4. Choose **Get more add-ons**.
5. On the **Select add-ons** page, choose **Amazon GuardDuty EKS Runtime Monitoring**.
6. GuardDuty recommends choosing the latest and default agent **Version**.
7. On the **Configure selected add-on settings** page, use the default settings. If the **Status** of your EKS add-on is **Requires activation**, choose **Activate GuardDuty**. This action will open the GuardDuty console to configure Runtime Monitoring for your accounts.
8. After you've configured Runtime Monitoring for your accounts, switch back to the Amazon EKS console. The **Status** of your EKS add-on should have changed to **Ready to install**.
9. **(Optional) Providing EKS add-on configuration schema**

   For the add-on **Version**, if you choose **v1.5.0** or above, Runtime Monitoring supports configuring specific parameters of the GuardDuty agent. For information about parameter ranges, see Configure EKS add-on parameters.

   a. Expand **Optional configuration settings** to view the configurable parameters and their expected value and format.
   b. Set the parameters. The values must be in the range provided in Configure EKS add-on parameters.
   c. Choose **Save changes** to create the add-on based on the advanced configuration.
   d. For **Conflict resolution method**, the option that you choose will be used to resolve a conflict when you update the value of a parameter to a non-default value. For more information about the listed options, see resolveConflicts in the *Amazon EKS API Reference*.
10. Choose **Next**.
11. On the **Review and create** page, verify all the details, and choose **Create**.
12. Navigate back to the cluster details and choose the **Resources** tab.
13. You can view the new pods with the prefix **aws-guardduty-agent**.

API/CLI

You can configure the Amazon EKS add-on agent (`aws-guardduty-agent`) using either of the following options:

- Run CreateAddon for your account.
- **Note**

  For the add-on `version`, if you choose **v1.5.0 or above**, Runtime Monitoring supports configuring specific parameters of the GuardDuty agent. For more information, see Configure EKS add-on parameters.

  Use the following values for the request parameters:

  - For `addonName`, enter `aws-guardduty-agent`.

    You can use the following AWS CLI example when using configurable values supported for add-on versions `v1.5.0` or above. Make sure to replace the placeholder values highlighted in red and the associated `Example.json` with the configured values.

    ```
    aws eks create-addon --region us-east-1 --cluster-name myClusterName --addon-
    name aws-guardduty-agent --addon-version v1.11.0-eksbuild.2 --configuration-
    values 'file://example.json'
    ```

    Example Example.json

    ```
    {
     "priorityClassName": "aws-guardduty-agent.priorityclass-high",
     "dnsPolicy": "Default",
     "resources": {
      "requests": {
       "cpu": "237m",
       "memory": "512Mi"
      },
      "limits": {
       "cpu": "2000m",
       "memory": "2048Mi"
      }
     }
    ```

```
}
```

    - For information about supported `addonVersion`, see Kubernetes versions supported by GuardDuty security agent.
- Alternatively, you can use AWS CLI. For more information, see create-addon.

Private DNS names for VPC endpoint

By default, the security agent resolves and connects to the private DNS name of the VPC endpoint. For a non-FIPS endpoint, your private DNS will appear in the following format:

Non-FIPS endpoint – `guardduty-data.us-east-1.amazonaws.com`

The AWS Region, *`us-east-1`*, will change based on your Region.

Updating security agent manually for Amazon EKS resources

When you manage the GuardDuty security agent manually, you are responsible to update it for your account. For notification about new agent versions, you can subscribe to an RSS feed to GuardDuty security agent release versions.

You can update the security agent to the latest version to benefit from the added support and improvements. If your current agent version is reaching an end of standard support, then to continue using Runtime Monitoring (or EKS Runtime Monitoring), you must update to a next available or the latest agent version.

Prerequisite

Before you update the security agent version, make sure that the agent version that you're planning to use now, is compatible with your Kubernetes version. For more information, see Kubernetes versions supported by GuardDuty security agent.

Console

1. Open the Amazon EKS console at https://console.aws.amazon.com/eks/home#/clusters.
2. Choose your **Cluster name**.
3. Under the **Cluster info**, choose the **Add-ons** tab.
4. Under the **Add-ons** tab, select **GuardDuty EKS Runtime Monitoring**.

5. Choose **Edit** to update the agent details.
6. On the **Configure GuardDuty EKS Runtime Monitoring** page, update the details.
7. **(Optional) Updating Optional configuration settings**

   If your EKS add-on **Version** is *1.5.0* or above, you can also update the add-on configuration schema.

   a. Expand **Optional configuration settings** to view the configuration schema.
   b. Update the parameter values based on the range provided in Configure EKS add-on parameters.
   c. Choose **Save changes** to start the update.
   d. For **Conflict resolution method**, the option that you choose will be used to resolve a conflict when you update the value of a parameter to a non-default value. For more information about the listed options, see resolveConflicts in the *Amazon EKS API Reference*.

API/CLI

To update the GuardDuty security agent for your Amazon EKS clusters, see Updating an add-on.

> **Note**
>
> For the add-on `version`, if you choose **1.5.0 or above**, Runtime Monitoring supports configuring specific parameters of the GuardDuty agent. For information about parameter ranges, see Configure EKS add-on parameters.

You can use the following AWS CLI example when using configurable values supported for add-on versions *1.5.0 and above*. Make sure to replace the placeholder values highlighted in red and the associated `Example.json` with the configured values.

```
aws eks update-addon --region us-east-1 --cluster-name myClusterName --addon-name aws-guardduty-agent --addon-version v1.11.0-eksbuild.2 --configuration-values 'file://example.json'
```

Example Example.json

```
{
```

```
  "priorityClassName": "aws-guardduty-agent.priorityclass-high",
  "dnsPolicy": "Default",
  "resources": {
   "requests": {
    "cpu": "237m",
    "memory": "512Mi"
   },
   "limits": {
    "cpu": "2000m",
    "memory": "2048Mi"
   }
  }
 }
```

If your Amazon EKS add-on version is 1.5.0 or above, and you have configured the add-on schema, you can verify whether or not the values appear correctly for your cluster. For more information, see Verifying configuration schema updates.

Configure GuardDuty security agent (add-on) parameters for Amazon EKS

You can configure specific parameters of your GuardDuty security agent for Amazon EKS. This support is available for GuardDuty security agent version 1.5.0 and above. For information about latest add-on versions, see GuardDuty security agent versions for Amazon EKS resources.

Why should I update the security agent configuration schema

> Configuration schema for the GuardDuty security agent is the same across all containers within your Amazon EKS clusters. When the default values do not align with the associated workloads and instance size, consider configuring the CPU settings, memory settings, `PriorityClass`, and `dnsPolicy` settings. Regardless of how you manage the GuardDuty agent for your Amazon EKS clusters, you can configure or update the existing configuration of these parameters.

Automated agent configuration behavior with configured parameters

When GuardDuty manages the security agent (EKS add-on) on your behalf, it updates the add-on, as needed. GuardDuty will set the value of the configurable parameters to a default value. However, you can still update the parameters to a desired value. If this leads to a conflict, the default option to resolveConflicts is None.

Configurable parameters and values

For information about the steps to configure the add-on parameters, see:

- Installing GuardDuty security agent manually on Amazon EKS resources or
- Updating security agent manually for Amazon EKS resources

The following tables provide the ranges and values that you can use to deploy the Amazon EKS add-on manually or update the existing add-on settings.

CPU settings

<table>
<tr><th>Parameters</th><th>Default value</th><th>Configurable range</th></tr>
<tr><td>Requests</td><td>200m</td><td rowspan="2">Between 200m and 10000m, both inclusive</td></tr>
<tr><td>Limits</td><td>1000m</td></tr>
</table>

Memory settings

<table>
<tr><th>Parameters</th><th>Default value</th><th>Configurable range</th></tr>
<tr><td>Requests</td><td>256Mi</td><td rowspan="2">Between 256Mi and 20000Mi, both inclusive</td></tr>
<tr><td>Limits</td><td>1024Mi</td></tr>
</table>

`PriorityClass` settings

When GuardDuty creates an Amazon EKS add-on for you, the assigned `PriorityClass` is `aws-guardduty-agent.priorityclass`. This means that no action will be taken based on the priority of the agent pod. You can configure this add-on parameter by choosing one of the following `PriorityClass` options:

Configurable PriorityClass	preemptionPolicy value	preemptionPolicy description	Pod value
aws-guardduty-agent.priorityclass	Never	No action	1000000
aws-guardduty-agent.priorityclass-high	PreemptLowerPriority	Assigning this value will preempt a pod running with the priority value lower than the agent pod value.	100000000
system-cluster-critical [1]	PreemptLowerPriority		2000000000
system-node-critical [1]	PreemptLowerPriority		2000001000

[1] Kubernetes provides these two PriorityClass options – system-cluster-critical and system-node-critical. For more information, see PriorityClass in the *Kubernetes documentation*.

dnsPolicy settings

Choose one of the following DNS policy options that Kubernetes supports. When no configuration is specified, ClusterFirst is used as the default value.

- ClusterFirst
- ClusterFirstWithHostNet
- Default

For information about these policies, see Pod's DNS Policy in the *Kubernetes documentation*.

Verifying configuration schema updates

After you have configured the parameters, perform the following steps to verify that the configuration schema has been updated:

1. Open the Amazon EKS console at https://console.aws.amazon.com/eks/home#/clusters.
2. In the navigation pane, choose **Clusters**.
3. On the **Clusters** page, select the **Cluster name** for which you want to verify the updates.
4. Choose the **Resources** tab.
5. From the **Resource types** pane, under **Workloads**, choose **DaemonSets**.
6. Select **aws-guardduty-agent**.
7. On the **aws-guardduty-agent** page, choose **Raw view** to view the unformatted JSON response. Verify that the configurable parameters display the value that you provided.

After you verify, switch to the GuardDuty console. Select the corresponding AWS Region and view the coverage status for your Amazon EKS clusters. For more information, see Runtime coverage and troubleshooting for Amazon EKS clusters.

Validating VPC endpoint configuration

After you install the security agent manually or through GuardDuty automated configuration, you can use this document to validate that the VPC endpoint configuration. You can also use these steps after troubleshooting any runtime coverage issue for a resource type. You can ensure that the steps worked as expected and the coverage status would potentially show up as **Healthy**.

Use the following steps to validate that VPC endpoint configuration for your resource type is set up correctly in the VPC owner account:

1. Sign in to the AWS Management Console and open the Amazon VPC console at https://console.aws.amazon.com/vpc/.
2. In the navigation pane, under **Virtual private cloud**, choose **Your VPCs**.
3. On the **Your VPCs** page, choose **IPv4 CIDR** associated with your **VPC ID**.
4. In the navigation pane, under **Virtual private cloud**, choose **Endpoints**.
5. In the **Endpoints** table, select the row that has the **Service name** similar to **com.amazonaws.*us-east-1*.guardduty-data**. The Region (`us-east-1`) might be different for your endpoint.

6. A panel for endpoint details will appear. Under the **Security Groups** tab, select the associated **Group ID** link for more details.
7. In the **Security Groups** table, select the row that with the associated **Security group ID** to view the details.
8. Under the **Inbound rules** tab, ensure that there is an ingress policy with **Port range** as **443** and **Source** as the value copied from the **IPv4 CIDR**. Inbound rules control the incoming traffic that is allowed to reach the instance. The following image shows the inbound rules for a security group that is associated with the VPC used by the GuardDuty security agent.

   If you don't already have a security group that has an in-bound port 443 enabled, Create a security group in the *Amazon EC2 User Guide*.

   If there is an issue while restricting the in-bound permissions to your VPC (or cluster), provide the support to in-bound 443 port from any IP address (0.0.0.0/0).

The following list includes good to know items after you install or update the security agent.

Assess runtime coverage

The next step after installing or updating your security agent is to assess runtime coverage of your resources. If the runtime coverage status is **Unhealthy**, then you must troubleshoot the issue. For more information, see Runtime coverage issues and troubleshooting.

If the status of the runtime coverage shows as **Healthy**, it indicates that Runtime Monitoring is able to collect and receive runtime events. For a list of these events, see Collected runtime event types.

Private DNS name for endpoint

After you install the GuardDuty security agent for your resources, by default, it will resolve and connect to the private DNS name of the VPC endpoint. For a non-FIPS endpoint, the private DNS will appear in the following format:

```
guardduty-data.us-east-1.amazonaws.com
```

The AWS Region, *us-east-1*, will change based on your Region.

A host may get installed with two security agents

When working with GuardDuty security agent for an Amazon EC2 instance, you might install and use the agent on the underlying host within an Amazon EKS cluster. If you had already deployed a security agent on that EKS cluster, the same host could have two security agents running on it at the same time. For information about how GuardDuty works in this scenario, see Security agents on same host.

Reviewing runtime coverage statistics and troubleshooting issues

After you enable Runtime Monitoring and the GuardDuty security agent gets deployed to your resource, GuardDuty provides coverage statistics for the corresponding resource type and individual coverage status for the resources that belong to your account. Coverage status is determined by making sure that you have enabled Runtime Monitoring, your Amazon VPC endpoint has been created, and the GuardDuty security agent for the corresponding resource has been deployed. A **Healthy** coverage status indicates that when there is a runtime event related to your resource, GuardDuty is able to receive the said runtime event through the Amazon VPC endpoint, and monitor the behavior. If there was an issue at the time of configuring Runtime Monitoring, creating an Amazon VPC endpoint, or deploying the GuardDuty security agent, the coverage status appears as **Unhealthy**. When the coverage status is unhealthy, GuardDuty will not be able to receive or monitor the runtime behavior of the corresponding resource, or generate any Runtime Monitoring findings.

The following topics will help you review coverage statistics, configure EventBridge notifications, and troubleshoot the coverage issues for a specific resource type.

Contents

- Runtime coverage and troubleshooting for Amazon EC2 instance
- Runtime coverage and troubleshooting for Amazon ECS clusters
- Runtime coverage and troubleshooting for Amazon EKS clusters

Runtime coverage and troubleshooting for Amazon EC2 instance

For an Amazon EC2 resource, the runtime coverage is evaluated at the instance level. Your Amazon EC2 instances can run multiple types of applications and workloads amongst others in your AWS environment. This feature also supports Amazon ECS managed Amazon EC2 instances and if you have Amazon ECS clusters running on an Amazon EC2 instance, the coverage issues at the instance level will show up under Amazon EC2 runtime coverage.

Topics

- Reviewing coverage statistics
- Coverage status change with EventBridge notifications
- Troubleshooting Amazon EC2 runtime coverage issues

Reviewing coverage statistics

The coverage statistics for the Amazon EC2 instances associated with your own accounts or your member accounts is the percentage of the healthy EC2 instances over all EC2 instances in the selected AWS Region. The following equation represents this as:

*(Healthy instances/All instances)*100*

If you have also deployed the GuardDuty security agent for your Amazon ECS clusters, then any instance level coverage issue associated with Amazon ECS clusters running on an Amazon EC2 instance will appear as an Amazon EC2 instance runtime coverage issue.

Choose one of the access methods to review the coverage statistics for your accounts.

Console

- Sign in to the AWS Management Console and open the GuardDuty console at https://console.aws.amazon.com/guardduty/.
- In the navigation pane, choose **Runtime Monitoring**.
- Choose the **Runtime coverage** tab.
- Under the **EC2 instance runtime coverage** tab, you can view the coverage statistics aggregated by the coverage status of each Amazon EC2 instance that is available in the **Instances list** table.
  - You can filter the **Instance list** table by the following columns:
    - **Account ID**
    - **Agent management type**
    - **Agent version**
    - **Coverage status**
    - **Instance ID**
    - **Cluster ARN**

- If any of your EC2 instances have the **Coverage status** as **Unhealthy**, the **Issue** column includes additional information about the reason for the **Unhealthy** status.

API/CLI

- Run the [ListCoverage](#) API with your own valid detector ID, current Region, and service endpoint. You can filter and sort the instance list using this API.
  - You can change the example `filter-criteria` with one of the following options for `CriterionKey`:
    - `ACCOUNT_ID`
    - `RESOURCE_TYPE`
    - `COVERAGE_STATUS`
    - `AGENT_VERSION`
    - `MANAGEMENT_TYPE`
    - `INSTANCE_ID`
    - `CLUSTER_ARN`
  - When the `filter-criteria` includes `RESOURCE_TYPE` as **EC2**, Runtime Monitoring doesn't support the use of **ISSUE** as the `AttributeName`. If you use it, the API response will result in `InvalidInputException`.

    You can change the example `AttributeName` in `sort-criteria` with the following options:
    - `ACCOUNT_ID`
    - `COVERAGE_STATUS`
    - `INSTANCE_ID`
    - `UPDATED_AT`
  - You can change the *`max-results`* (up to 50).
  - To find the `detectorId` for your account and current Region, see the **Settings** page in the https://console.aws.amazon.com/guardduty/ console, or run the [ListDetectors](#) API.

```
aws guardduty --region us-east-1 list-coverage --detector-
id 12abc34d567e8fa901bc2d34e56789f0 --sort-criteria '{"AttributeName":
 "EKS_CLUSTER_NAME", "OrderBy": "DESC"}' --filter-criteria
```

```
 '{"FilterCriterion":[{"CriterionKey":"ACCOUNT_ID", "FilterCondition":
{"EqualsValue":"111122223333"}}] }'  --max-results 5
```

- Run the GetCoverageStatistics API to retrieve coverage aggregated statistics based on the statisticsType.
  - You can change the example statisticsType to one of the following options:
    - COUNT_BY_COVERAGE_STATUS – Represents coverage statistics for EKS clusters aggregated by coverage status.
    - COUNT_BY_RESOURCE_TYPE – Coverage statistics aggregated based on the type of AWS resource in the list.
    - You can change the example filter-criteria in the command. You can use the following options for CriterionKey:
      - ACCOUNT_ID
      - RESOURCE_TYPE
      - COVERAGE_STATUS
      - AGENT_VERSION
      - MANAGEMENT_TYPE
      - INSTANCE_ID
      - CLUSTER_ARN
  - To find the detectorId for your account and current Region, see the **Settings** page in the https://console.aws.amazon.com/guardduty/ console, or run the ListDetectors API.

```
aws guardduty --region us-east-1 get-coverage-statistics --detector-
id 12abc34d567e8fa901bc2d34e56789f0 --statistics-type COUNT_BY_COVERAGE_STATUS
 --filter-criteria '{"FilterCriterion":[{"CriterionKey":"ACCOUNT_ID",
 "FilterCondition":{"EqualsValue":"123456789012"}}] }'
```

If the coverage status of your EC2 instance is **Unhealthy**, see Troubleshooting Amazon EC2 runtime coverage issues.

Coverage status change with EventBridge notifications

The coverage status of your Amazon EC2 instance might appear as **Unhealthy**. To know when the coverage status changes, we recommend you to monitor the coverage status periodically, and troubleshoot if the status becomes **Unhealthy**. Alternatively, you can create an Amazon

EventBridge rule to receive a notification when the coverage status changes from either **Unhealthy** to **Healthy** or otherwise. By default, GuardDuty publishes this in the EventBridge bus for your account.

Sample notification schema

In an EventBridge rule, you can use the pre-defined sample events and event patterns to receive coverage status notification. For more information about creating an EventBridge rule, see Create rule in the *Amazon EventBridge User Guide*.

Additionally, you can create a custom event pattern by using the following example notification schema. Make sure to replace the values for your account. To get notified when the coverage status of your Amazon EC2 instance changes from `Healthy` to `Unhealthy`, the `detail-type` should be *`GuardDuty Runtime Protection Unhealthy`*. To get notified when the coverage status changes from `Unhealthy` to `Healthy`, replace the value of `detail-type` with *`GuardDuty Runtime Protection Healthy`*.

```
{
  "version": "0",
  "id": "event ID",
  "detail-type": "GuardDuty Runtime Protection Unhealthy",
  "source": "aws.guardduty",
  "account": "AWS account ID",
  "time": "event timestamp (string)",
  "region": "AWS Region",
  "resources": [
       ],
  "detail": {
    "schemaVersion": "1.0",
    "resourceAccountId": "string",
    "currentStatus": "string",
    "previousStatus": "string",
    "resourceDetails": {
        "resourceType": "EC2",
        "ec2InstanceDetails": {
          "instanceId":"",
          "instanceType":"",
          "clusterArn": "",
          "agentDetails": {
            "version":""
          },
          "managementType":""
```

```
        }
    },
    "issue": "string",
    "lastUpdatedAt": "timestamp"
  }
}
```

Troubleshooting Amazon EC2 runtime coverage issues

If the coverage status of your Amazon EC2 instance is **Unhealthy**, you can view the reason under the **Issue** column.

If your EC2 instance is associated with an EKS cluster and the security agent for EKS was installed either manually or through automated agent configuration, then to troubleshoot the coverage issue, see Runtime coverage and troubleshooting for Amazon EKS clusters.

The following table lists the issue types and the corresponding troubleshooting steps.

Issue type	Issue message	Troubleshooting steps
No Agent Reporting	Waiting for SSM notification	Receiving the SSM notificat ion may take a few minutes. Make sure that the Amazon EC2 instance is SSM managed. For more information, see the steps under *Method 1 - By using AWS Systems Manager* in Installing the security agent manually.
	(Empty on purpose)	If you are managing the GuardDuty security agent manually, make sure that you followed the steps under Managing security agent manually for Amazon EC2 resource.

Issue type	Issue message	Troubleshooting steps
		If you've enabled automated agent configuration: • Your EC2 instance is SSM managed. • View the status of your security agent periodically. For more information, see [Validating GuardDuty security agent installation status]().
		Validate that the VPC endpoint for your Amazon EC2 instance is correctly configured. For more information, see [Validating VPC endpoint configuration]().
		If your organization has a service control policy (SCP), validate that permissions boundary is not restricting the `guardduty:SendSecurityTelemetry` permission. For more information, see [Validating your organization service control policy in a multi-account environment]().

Issue type	Issue message	Troubleshooting steps
	Agent disconnected	• View the status of your security agent. For more information, see [Validating GuardDuty security agent installation status](). • View the security agent logs to identify the potential root cause. The logs provide detailed errors that you can use to troubleshoot the issue yourself. The log files are available under `/var/log/amzn-guardduty-agent/` . Do `sudo journalctl -u amazon-guardduty-agent` .
Agent Not Provisioned	Instances with exclusion tags are excluded from Runtime Monitoring.	GuardDuty doesn't receive runtime events from Amazon EC2 instances that are launched with the exclusion tag `GuardDutyManaged :false`. To receive runtime events from this Amazon EC2 instance, remove the exclusion tag.

Issue type	Issue message	Troubleshooting steps
	Kernel version is lower than the supported version.	For information about supported kernel versions across OS distributions, see Validate architectural requirements for Amazon EC2 instances.
	Kernel version is higher than the supported version.	For information about supported kernel versions across OS distributions, see Validate architectural requirements for Amazon EC2 instances.

Issue type	Issue message	Troubleshooting steps
	Unable to retrieve instance identity document.	Follow these steps: 1. Confirm that your resource is an Amazon EC2 instance, and not a hybrid non-EC2 instance. 2. Confirm that the Instance Metadata Service (IMDS) is enabled. To do this, see Configure Instance Metadata Service options in the *Amazon EC2 User Guide*. 3. Verify that the instance identity document exists. To do this, see Retrieve the instance identity document in the *Amazon EC2 User Guide*. 4. If the instance identity document still doesn't exists, then restart the instance. The instance identity document is generated when the instance is stopped and started, restarted, or launched.

Issue type	Issue message	Troubleshooting steps
SSM Association Creation Failed	GuardDuty SSM association already exists in your account	1. Delete the existing association manually. For more information, see Deleting associations in the *AWS Systems Manager User Guide*. 2. After you delete the association, disable and then re-enable the GuardDuty automated agent configuration for Amazon EC2.
	Your account has too many SSM associations	Choose **one** of the following two options: • Delete any unused SSM associations. For more information, see Deleting associations in the *AWS Systems Manager User Guide*. • Check if your account is eligible for a quota increase. For more information, see Systems Manager Service quotas in the *AWS General Reference*.
SSM Association Updation Failed	GuardDuty SSM association does not exist in your account	GuardDuty SSM association is not present in your account. Disable and then re-enable Runtime Monitoring.

Issue type	Issue message	Troubleshooting steps
SSM Association Deletion Failed	GuardDuty SSM association does not exist in your account	The SSM association is not present in your account. If the SSM association was deleted intentionally, then no action is needed.

Issue type	Issue message	Troubleshooting steps
SSM Instance Association Execution Failed	Architectural requirements or other prerequisites are not met.	For information about verified operating system distributions, see [Prerequisites for Amazon EC2 instance support](). If you still experience this issue, the following steps will help you identify and potentially resolve the issue: 1. Open the AWS Systems Manager console at https://console.aws.amazon.com/systems-manager/. 2. In the navigation pane, under **Node management**, select **State Manager**. 3. Filter by **Document Name** property and enter **AmazonGuardDuty-ConfigureRuntimeMonitoringSsmPlugin**. 4. Select the corresponding association ID and view its **Execution history**. 5. Using the execution history, view the failures, identify the potential root cause, and try to resolve it.

Issue type	Issue message	Troubleshooting steps
VPC Endpoint Creation Failed	VPC endpoint creation not supported for shared VPC *vpcId*	Runtime Monitoring supports the use of a shared VPC within an organization. For more information, see Using shared VPC with Runtime Monitoring.
	Only when using shared VPC with automated agent configuration Owner account ID *111122223333* for shared VPC *vpcId* doesn't have either Runtime Monitoring, automated agent configura tion, or both, enabled	The shared VPC owner account must enable Runtime Monitoring and automated agent configuration for at least one resource type (Amazon EKS or Amazon ECS (AWS Fargate)). For more information, see Prerequis ites specific to GuardDuty Runtime Monitoring.
	Enabling private DNS requires both `enableDnsSupport` and `enableDnsHostnames` VPC attributes set to `true` for *vpcId* (Service: Ec2, Status Code:400, Request ID: *a1b2c3d4-5678-90ab-cdef-EXAMPLE11111*).	Ensure that the following VPC attributes are set to `true` – `enableDnsSupport` and `enableDnsHostnames` . For more information, see DNS attributes in your VPC. If you're using Amazon VPC Console at https://console.a ws.amazon.com/vpc/ to create the Amazon VPC, make sure to select both **Enable DNS hostnames** and **Enable DNS resolution**. For more information, see VPC configuration options.

Issue type	Issue message	Troubleshooting steps
Shared VPC Endpoint Deletion Failed	Shared VPC endpoint deletion not allowed for account ID *111122223333* , shared VPC *vpcId*, owner account ID *555555555555* .	**Potential steps:** • Disabling the Runtime Monitoring status of the shared VPC participant account doesn't impact the shared VPC endpoint policy and the security group that exists in the owner account. To delete the shared VPC endpoint and security group, you must disable Runtime Monitoring or automated agent configura tion status in the shared VPC owner account. • The shared VPC participa nt account can't delete the shared VPC endpoint and security group hosted in the shared VPC owner account.
Agent not reporting	(Empty on purpose)	The issue type has reached end of support. If you continue experiencing this issue and not already done so, enable GuardDuty automated agent for Amazon EC2. If the issue still persists, consider disabling Runtime Monitoring for a few minutes and then enable it again.

Runtime coverage and troubleshooting for Amazon ECS clusters

The runtime coverage for Amazon ECS clusters includes the tasks running on AWS Fargate and Amazon ECS container instances[1].

For an Amazon ECS cluster that runs on Fargate, the runtime coverage is assessed at the task level. The ECS clusters runtime coverage includes those Fargate tasks that have started running after you have enabled Runtime Monitoring and automated agent configuration for Fargate (ECS only). By default, a Fargate task is immutable. GuardDuty will not be able to install the security agent to monitor containers on already running tasks. To include such a Fargate task, you must stop and start the task again. Make sure to check if the associated service is supported.

For information about Amazon ECS container, see Capacity creation.

Contents

- Reviewing coverage statistics
- Coverage status change with EventBridge notifications
- Troubleshooting Amazon ECS-Fargate runtime coverage issues

Reviewing coverage statistics

The coverage statistics for the Amazon ECS resources associated with your own account or your member accounts is the percentage of the healthy Amazon ECS clusters over all the Amazon ECS clusters in the selected AWS Region. This includes the coverage for Amazon ECS clusters associated with both Fargate and Amazon EC2 instances. The following equation represents this as:

*(Healthy clusters/All clusters)*100*

Considerations

- The coverage statistics for the ECS cluster include the coverage status of the Fargate tasks or ECS container instances associated with that ECS cluster. The coverage status of the Fargate tasks include tasks that either are in running state or have recently finished running.
- In the **ECS clusters runtime coverage** tab, the **Container instances covered** field indicates the coverage status of the container instances associated with your Amazon ECS cluster.

  If your Amazon ECS cluster contains only Fargate tasks, the count appears as **0/0**.

- If your Amazon ECS cluster is associated with an Amazon EC2 instance that doesn't have a security agent, the Amazon ECS cluster will also have an **Unhealthy** coverage status.

  To identify and troubleshoot the coverage issue for the associated Amazon EC2 instance, see [Troubleshooting Amazon EC2 runtime coverage issues](#) for Amazon EC2 instances.

Choose one of the access methods to review the coverage statistics for your accounts.

Console

- Sign in to the AWS Management Console and open the GuardDuty console at https://console.aws.amazon.com/guardduty/.
- In the navigation pane, choose **Runtime Monitoring**.
- Choose the **Runtime coverage** tab.
- Under the **ECS clusters runtime coverage** tab, you can view the coverage statistics aggregated by the coverage status of each Amazon ECS cluster that is available in the **Clusters list** table.
  - You can filter the **Cluster list** table by the following columns:
    - **Account ID**
    - **Cluster Name**
    - **Agent management type**
    - **Coverage status**
- If any of your Amazon ECS clusters have the **Coverage status** as **Unhealthy**, the **Issue** column includes additional information about the reason for the **Unhealthy** status.

  If you Amazon ECS clusters are associated with an Amazon EC2 instance, navigate to the **EC2 instance runtime coverage** tab and filter by the **Cluster name** field to view the associated **Issue**.

API/CLI

- Run the [ListCoverage](#) API with your own valid detector ID, current Region, and service endpoint. You can filter and sort the instance list using this API.
  - You can change the example `filter-criteria` with one of the following options for `CriterionKey`:

    - ACCOUNT_ID
    - ECS_CLUSTER_NAME
    - COVERAGE_STATUS
    - MANAGEMENT_TYPE
  - You can change the example AttributeName in sort-criteria with the following options:
    - ACCOUNT_ID
    - COVERAGE_STATUS
    - ISSUE
    - ECS_CLUSTER_NAME
    - UPDATED_AT

      The field gets updated only when either a new task gets created in the associated Amazon ECS cluster or there is change in the corresponding coverage status.
  - You can change the *max-results* (up to 50).
  - To find the detectorId for your account and current Region, see the **Settings** page in the https://console.aws.amazon.com/guardduty/ console, or run the ListDetectors API.

```
aws guardduty --region us-east-1 list-coverage --detector-
id 12abc34d567e8fa901bc2d34e56789f0 --sort-criteria '{"AttributeName":
 "ECS_CLUSTER_NAME", "OrderBy": "DESC"}' --filter-criteria
 '{"FilterCriterion":[{"CriterionKey":"ACCOUNT_ID", "FilterCondition":
{"EqualsValue":"111122223333"}}] }'  --max-results 5
```

- Run the GetCoverageStatistics API to retrieve coverage aggregated statistics based on the statisticsType.
  - You can change the example statisticsType to one of the following options:
    - COUNT_BY_COVERAGE_STATUS – Represents coverage statistics for ECS clusters aggregated by coverage status.
    - COUNT_BY_RESOURCE_TYPE – Coverage statistics aggregated based on the type of AWS resource in the list.
    - You can change the example filter-criteria in the command. You can use the following options for CriterionKey:
      - ACCOUNT_ID

    - `ECS_CLUSTER_NAME`
    - `COVERAGE_STATUS`
    - `MANAGEMENT_TYPE`
    - `INSTANCE_ID`
- To find the `detectorId` for your account and current Region, see the **Settings** page in the https://console.aws.amazon.com/guardduty/ console, or run the ListDetectors API.

```
aws guardduty --region us-east-1 get-coverage-statistics --detector-
id 12abc34d567e8fa901bc2d34e56789f0 --statistics-type COUNT_BY_COVERAGE_STATUS
 --filter-criteria '{"FilterCriterion":[{"CriterionKey":"ACCOUNT_ID",
 "FilterCondition":{"EqualsValue":"123456789012"}}] }'
```

For more information about coverage issues, see Troubleshooting Amazon ECS-Fargate runtime coverage issues.

Coverage status change with EventBridge notifications

The coverage status of your Amazon ECS cluster might appear as **Unhealthy**. To know when the coverage status changes, we recommend you to monitor the coverage status periodically, and troubleshoot if the status becomes **Unhealthy**. Alternatively, you can create an Amazon EventBridge rule to receive a notification when the coverage status changes from either **Unhealthy** to **Healthy** or otherwise. By default, GuardDuty publishes this in the EventBridge bus for your account.

Sample notification schema

In an EventBridge rule, you can use the pre-defined sample events and event patterns to receive coverage status notification. For more information about creating an EventBridge rule, see Create rule in the *Amazon EventBridge User Guide*.

Additionally, you can create a custom event pattern by using the following example notification schema. Make sure to replace the values for your account. To get notified when the coverage status of your Amazon ECS cluster changes from `Healthy` to `Unhealthy`, the `detail-type` should be *`GuardDuty Runtime Protection Unhealthy`*. To get notified when the coverage status changes from `Unhealthy` to `Healthy`, replace the value of `detail-type` with *`GuardDuty Runtime Protection Healthy`*.

```
{
```

```
  "version": "0",
  "id": "event ID",
  "detail-type": "GuardDuty Runtime Protection Unhealthy",
  "source": "aws.guardduty",
  "account": "AWS account ID",
  "time": "event timestamp (string)",
  "region": "AWS Region",
  "resources": [
       ],
  "detail": {
    "schemaVersion": "1.0",
    "resourceAccountId": "string",
    "currentStatus": "string",
    "previousStatus": "string",
    "resourceDetails": {
        "resourceType": "ECS",
        "ecsClusterDetails": {
          "clusterName":"",
          "fargateDetails":{
            "issues":[],
            "managementType":""
          },
          "containerInstanceDetails":{
            "coveredContainerInstances":int,
            "compatibleContainerInstances":int
          }
        }
    },
    "issue": "string",
    "lastUpdatedAt": "timestamp"
  }
}
```

Troubleshooting Amazon ECS-Fargate runtime coverage issues

If the coverage status of your Amazon ECS cluster is **Unhealthy**, you can view the reason under the **Issue** column.

The following table provides the recommended troubleshooting steps for Fargate (Amazon ECS only) issues. For information about Amazon EC2 instance coverage issues, see Troubleshooting Amazon EC2 runtime coverage issues for Amazon EC2 instances.

Issue type	Extra information	Recommended troubleshooting steps
Agent not reporting	Agent not reporting for tasks in `TaskDefinition - '`*`TASK_DEFINITION`*`'`	Validate that the VPC endpoint for your Amazon ECS cluster's task is correctly configured. For more information, see [Validating VPC endpoint configuration](). If your organization has a service control policy (SCP), validate that permissions boundary is not restricting the `guardduty:SendSecurityTelemetry` permission. For more information, see [Validating your organization service control policy in a multi-account environment]().
	`VPC_ISSUE` `; for task in TaskDefinition - '`*`TASK_DEFINITION`*`'`	View the VPC issue details in the extra information.
Agent exited	ExitCode: `EXIT_CODE` for tasks in `TaskDefinition - '`*`TASK_DEFINITION`*`'`	View the issue details in the extra information.
	Reason: *`REASON`* for tasks in `TaskDefinition - '`*`TASK_DEFINITION`*`'`	
	ExitCode: `EXIT_CODE` with reason: '*`EXIT_CODE`*' for tasks in `TaskDefinition - '`*`TASK_DEFINITION`*`'`	

Issue type	Extra information	Recommended troubleshooting steps
	Agent exited: Reason: `CannotPullContainerError` : pull image manifest has been retried...	In this scenario, GuardDuty is potentially unable to pull the sidecar container image. Your task will continue to run but GuardDuty can't detect potential threats. Perform the following troubleshooting steps one at a time to check if it helps resolving the coverage issue: • **Permissions:** Ensure your task execution role has the required ECR permissions listed in [Permissions requirements](). • **Network connectivity:** Verify that your Fargate tasks can reach ECR either through public internet access or properly configured VPC endpoints as described in [Network connectivity requirements](). • **Security group configuration:** Check that your security group allows outbound access to the S3 managed prefix list on port 443 as explained in [Security group configuration](). • Run the [AWSSupport-TroubleshootECSTaskFailed]()

Issue type	Extra information	Recommended troubleshooting steps
		ToStart runbook in your cluster's Region to identify specific issues. • View tasks logs for error messages. For information about how to do this, see Viewing Amazon ECS container agent logs in the *Amazon Elastic Container Service Developer Guide*. For common errors and troubleshooting, see Amazon ECS troubleshooting in the *Amazon Elastic Container Service Developer Guide*. These three components (permissions, network connectivity, and security group configuration) are independent but all necessary for successfully downloading the GuardDuty container image from Amazon ECR. If the issue persists, see My AWS Step Functions workflow is failing unexpectedly.

Issue type	Extra information	Recommended troubleshooting steps
VPC Endpoint Creation Failed	Enabling private DNS requires both `enableDnsSupport` and `enableDnsHostnames` VPC attributes set to `true` for *`vpcId`* (Service: EC2, Status Code:400, Request ID: *`a1b2c3d4-5678-90ab-cdef-EXAMPLE11111`*).	Ensure that the following VPC attributes are set to `true` – `enableDnsSupport` and `enableDnsHostnames` . For more information, see DNS attributes in your VPC. If you're using Amazon VPC Console at https://console.aws.amazon.com/vpc/ to create the Amazon VPC, make sure to select both **Enable DNS hostnames** and **Enable DNS resolution**. For more information, see VPC configuration options.
Agent not provisioned	Unsupported invocation by *`SERVICE`* for task(s) in `TaskDefinition` - '*`TASK_DEFINITION`* '	This task was invoked by a *`SERVICE`* that is not supported.
	Unsupported CPU architecture '*`TYPE`*' for task(s) in `TaskDefinition` - '*`TASK_DEFINITION`* '	This task is running on an unsupported CPU architecture. For information about supported CPU architectures, see Validating architectural requirements.

Issue type	Extra information	Recommended troubleshooting steps
	`TaskExecutionRole` missing from `TaskDefinition - ' TASK_DEFINITION '`	The ECS task execution role is missing. For information about providing task execution role and required permissions, see Prerequisites for container image access.
	Missing network configuration '*CONFIGURATION_DETAILS* ' for task(s) in `TaskDefinition - 'TASK_DEFINITION '`	Network configuration issues may show up because of missing VPC configuration, or missing or empty subnets. Validate that your network configuration is correct. For more information, see Prerequisites for container image access. For more information, see Amazon ECS task definition parameters in the *Amazon Elastic Container Service Developer Guide*.

Issue type	Extra information	Recommended troubleshooting steps
	Tasks started when clusters had exclusion tag are excluded from Runtime Monitoring. Impacted task ID(s): '*TASK_ID*	When you change the predefined GuardDuty tag from `GuardDuty Managed -true` to `GuardDuty Managed -false`, GuardDuty will not receive the runtime events for this Amazon ECS cluster. Update the tag to `GuardDutyManaged -true` and then relaunch the task.
	Services deployed when clusters had exclusion tag are excluded from Runtime Monitoring. Impacted service name(s): '*SERVICE_NAME* '	When services deployed with the exclusion tag `GuardDuty Managed -false`, GuardDuty will not receive runtime events for this Amazon ECS cluster. Update the tag to `GuardDutyManaged -true` and then redeploy the service.
	Tasks started before enabling Automated Agent Configuration are not covered. Impacted task ID(s): '*TASK_ID*'	When cluster contains a task that launched before enabling the Automated agent configuration for Amazon ECS, then GuardDuty will be unable to protect this. Relaunch the task for it to be monitored by GuardDuty.

Issue type	Extra information	Recommended troubleshooting steps
	Services deployed before enabling Automated Agent Configuration are not covered. Impacted service name(s): '*SERVICE_NAME* '	When services are deployed before enabling Automated agent configuration for Amazon ECS, GuardDuty will not receive runtime events for ECS clusters.
	Service '*SERVICE_NAME* ' requires a new deployment to fix/troubleshoot. Refer documentation, Impacted service name(s): '*SERVICE_NAME* '	A service that started before enabling Runtime Monitoring is not supported. You can either restart the service or update the service with `forceNewDeployment` option by following the steps under Updating an Amazon ECS service using the console in the *Amazon Elastic Container Service Developer Guide*. Alternatively, you can also use the steps under UpdateService in the *Amazon Elastic Container Service API Reference*.
	Tasks started before enabling Runtime Monitoring require a relaunch. Impacted task ID(s): '*TASK_ID_1* '	In Amazon ECS, the tasks are immutable. To assess the runtime behavior or a running AWS Fargate task, make sure that Runtime Monitoring is already enabled, and then restart the task for GuardDuty to add the container sidecar.

Issue type	Extra information	Recommended troubleshooting steps
Others	Unidentified issue, for tasks in `TaskDefinition - 'TASK_DEFINITION '`	Use the following questions to identify the root cause of the issue: • Did the task start before you enabled Runtime Monitoring? In Amazon ECS, the tasks are immutable. To assess the runtime behavior of a running Fargate task, make sure that Runtime Monitoring is already enabled, and then restart the task for GuardDuty to add the container sidecar. • Is this task part of a service deployment that started before you enabled Runtime Monitoring? If yes, you can either restart the service or update the service with `forceNewDeployment` by using the steps in Updating a service. You can also use UpdateService or AWS CLI. • Did the task launch after excluding the ECS cluster from Runtime Monitoring?

Issue type	Extra information	Recommended troubleshooting steps
		When you change the pre-defined GuardDuty tag from `GuardDuty Managed -true` to `GuardDuty Managed -false`, GuardDuty will not receive the runtime events for the ECS cluster. • Does your service contain a task that has an old format of `taskArn`? GuardDuty Runtime Monitoring doesn't support the coverage for tasks that have the old format of `taskArn`. For information about Amazon Resource Names (ARNs) for Amazon ECS resources, see [Amazon Resource Names (ARNs) and IDs]().

Runtime coverage and troubleshooting for Amazon EKS clusters

After you enable Runtime Monitoring and install the GuardDuty security agent (add-on) for EKS either manually or through automated agent configuration, you can start assessing the coverage for your EKS clusters.

Contents

- Reviewing coverage statistics
- Coverage status change with EventBridge notifications
- Troubleshooting Amazon EKS runtime coverage issues

Reviewing coverage statistics

The coverage statistics for the EKS clusters associated with your own accounts or your member accounts is the percentage of the healthy EKS clusters over all EKS clusters in the selected AWS Region. The following equation represents this as:

*(Healthy clusters/All clusters)*100*

Choose one of the access methods to review the coverage statistics for your accounts.

Console

- Sign in to the AWS Management Console and open the GuardDuty console at https://console.aws.amazon.com/guardduty/.
- In the navigation pane, choose **Runtime Monitoring**.
- Choose the **EKS clusters runtime coverage** tab.
- Under the **EKS clusters runtime coverage** tab, you can view the coverage statistics aggregated by the coverage status that is available in the **Clusters list** table.
  - You can filter the **Clusters list** table by the following columns:
    - **Cluster name**
    - **Account ID**
    - **Agent management type**
    - **Coverage status**
    - **Add-on version**
- If any of your EKS clusters have the **Coverage status** as **Unhealthy**, the **Issue** column may include additional information about the reason for the **Unhealthy** status.

API/CLI

- Run the ListCoverage API with your own valid detector ID, Region, and service endpoint. You can filter and sort the cluster list using this API.

  - You can change the example `filter-criteria` with one of the following options for `CriterionKey`:
    - `ACCOUNT_ID`
    - `CLUSTER_NAME`
    - `RESOURCE_TYPE`
    - `COVERAGE_STATUS`
    - `ADDON_VERSION`
    - `MANAGEMENT_TYPE`
  - You can change the example `AttributeName` in `sort-criteria` with the following options:
    - `ACCOUNT_ID`
    - `CLUSTER_NAME`
    - `COVERAGE_STATUS`
    - `ISSUE`
    - `ADDON_VERSION`
    - `UPDATED_AT`
  - You can change the *`max-results`* (up to 50).
  - To find the `detectorId` for your account and current Region, see the **Settings** page in the https://console.aws.amazon.com/guardduty/ console, or run the ListDetectors API.

```
aws guardduty --region us-east-1 list-coverage --detector-id 12abc34d567e8fa901bc2d34e56789f0 --sort-criteria '{"AttributeName": "EKS_CLUSTER_NAME", "OrderBy": "DESC"}' --filter-criteria '{"FilterCriterion":[{"CriterionKey":"ACCOUNT_ID", "FilterCondition":{"EqualsValue":"111122223333"}}] }'  --max-results 5
```

- Run the GetCoverageStatistics API to retrieve coverage aggregated statistics based on the `statisticsType`.
  - You can change the example `statisticsType` to one of the following options:
    - `COUNT_BY_COVERAGE_STATUS` – Represents coverage statistics for EKS clusters aggregated by coverage status.
    - `COUNT_BY_RESOURCE_TYPE` – Coverage statistics aggregated based on the type of AWS resource in the list.

- You can change the example `filter-criteria` in the command. You can use the following options for `CriterionKey`:
  - `ACCOUNT_ID`
  - `CLUSTER_NAME`
  - `RESOURCE_TYPE`
  - `COVERAGE_STATUS`
  - `ADDON_VERSION`
  - `MANAGEMENT_TYPE`
- To find the `detectorId` for your account and current Region, see the **Settings** page in the https://console.aws.amazon.com/guardduty/ console, or run the ListDetectors API.

```
aws guardduty --region us-east-1 get-coverage-statistics --detector-
id 12abc34d567e8fa901bc2d34e56789f0 --statistics-type COUNT_BY_COVERAGE_STATUS
 --filter-criteria '{"FilterCriterion":[{"CriterionKey":"ACCOUNT_ID",
 "FilterCondition":{"EqualsValue":"123456789012"}}] }'
```

If the coverage status of your EKS cluster is **Unhealthy**, see Troubleshooting Amazon EKS runtime coverage issues.

Coverage status change with EventBridge notifications

The coverage status of an EKS cluster in your account may show up as **Unhealthy**. To detect when the coverage status becomes **Unhealthy**, we recommend you monitor the coverage status periodically and troubleshoot, if the status is **Unhealthy**. Alternatively, you can create an Amazon EventBridge rule to notify you when the coverage status changes from either `Unhealthy` to `Healthy` or otherwise. By default, GuardDuty publishes this in the EventBridge bus for your account.

Sample notification schema

In an EventBridge rule, you can use the pre-defined sample events and event patterns to receive coverage status notification. For more information about creating an EventBridge rule, see Create rule in the *Amazon EventBridge User Guide*.

Additionally, you can create a custom event pattern by using the following example notification schema. Make sure to replace the values for your account. To get notified when the coverage status of your Amazon EKS cluster changes from `Healthy` to `Unhealthy`, the `detail-type` should be *GuardDuty Runtime Protection Unhealthy*. To get notified when the coverage status changes from Unhealthy to Healthy, replace the value of detail-type with *GuardDuty Runtime Protection Healthy*.

```
{
  "version": "0",
  "id": "event ID",
  "detail-type": "GuardDuty Runtime Protection Unhealthy",
  "source": "aws.guardduty",
  "account": "AWS account ID",
  "time": "event timestamp (string)",
  "region": "AWS Region",
  "resources": [
       ],
  "detail": {
    "schemaVersion": "1.0",
    "resourceAccountId": "string",
    "currentStatus": "string",
    "previousStatus": "string",
    "resourceDetails": {
        "resourceType": "EKS",
        "eksClusterDetails": {
            "clusterName": "string",
            "availableNodes": "string",
             "desiredNodes": "string",
             "addonVersion": "string"
         }
    },
    "issue": "string",
    "lastUpdatedAt": "timestamp"
  }
}
```

Troubleshooting Amazon EKS runtime coverage issues

If the coverage status for your EKS cluster is Unhealthy, you can view the corresponding error either under the **Issue** column in the GuardDuty console, or by using the CoverageResource data type.

When working with inclusion or exclusion tags for monitoring your EKS clusters selectively, it may take some time for the tags to sync. This may impact the coverage status of the associated EKS cluster. You can try removing and adding the corresponding tag (inclusion or exclusion) again. For more information, see Tagging your Amazon EKS resources in the Amazon EKS User Guide.

The structure of a coverage issue is `Issue type:Extra information`. Typically, the issues will have an optional *Extra information* that may include specific client-side exception or description about the issue. Based on *Extra information*, the following tables provide the recommended steps to troubleshoot the coverage issues for your EKS clusters.

Issue type (prefix)	Extra information	Recommended troubleshooting steps
Addon Creation Failed	Addon `aws-guardduty-agent` is not compatible with current cluster version of cluster *`ClusterName`* . Addon specified is not supported.	Make sure that you're using one of those Kubernetes versions that support deploying the `aws-guardduty-agent` EKS add-on. For more information, see Kubernetes versions supported by GuardDuty security agent. For information about updating your Kubernetes version, see Updating an Amazon EKS cluster Kubernetes version.
Addon Creation Failed Addon Updation Failed Addon Status Unhealthy	EKS Addon issue - `AddonIssueCode : AddonIssueMessage`	For information about recommended steps for a specific add-on issue code, see Troubleshooting steps for Addon creation/updatation error with Addon issue code. For a list of addon issue codes that you might experience in this issue, see AddonIssue.

Issue type (prefix)	Extra information	Recommended troubleshooting steps
VPC Endpoint Creation Failed	VPC endpoint creation not supported for shared VPC *vpcId*	Runtime Monitoring now supports the use of a shared VPC within an organization. Make sure your accounts meet all the prerequisites. For more information, see Prerequisites for using shared VPC.
	Only when using shared VPC with automated agent configuration Owner account ID *111122223333* for shared VPC *vpcId* doesn't have either Runtime Monitoring, automated agent configuration, or both, enabled.	The shared VPC owner account must enable Runtime Monitoring and automated agent configuration for at least one resource type (Amazon EKS or Amazon ECS (AWS Fargate)). For more information, see Prerequisites specific to GuardDuty Runtime Monitoring.

Issue type (prefix)	Extra information	Recommended troubleshooting steps
	Enabling private DNS requires both `enableDnsSupport` and `enableDnsHostnames` VPC attributes set to `true` for *`vpcId`* (Service: Ec2, Status Code:400, Request ID: *`a1b2c3d4-5678-90ab-cdef-EXAMPLE11111`*).	Ensure that the following VPC attributes are set to `true` – `enableDnsSupport` and `enableDnsHostnames` . For more information, see [DNS attributes in your VPC](). If you're using Amazon VPC Console at [https://console.aws.amazon.com/vpc/]() to create the Amazon VPC, make sure to select both **Enable DNS hostnames** and **Enable DNS resolution**. For more information, see [VPC configuration options]().

Issue type (prefix)	Extra information	Recommended troubleshooting steps
Shared VPC Endpoint Deletion Failed	Shared VPC endpoint deletion not allowed for account ID *111122223333* , shared VPC *vpcId*, owner account ID *555555555555* .	**Potential steps:** • Disabling the Runtime Monitoring status of the shared VPC participant account doesn't impact the shared VPC endpoint policy and the security group that exists in the owner account. To delete the shared VPC endpoint and security group, you must disable Runtime Monitoring or automated agent configuration status in the shared VPC owner account. • The shared VPC participant account can't delete the shared VPC endpoint and security group hosted in the shared VPC owner account.
Local EKS clusters	EKS addons are not supported on local outpost clusters.	Not actionable. For more information, see Amazon EKS on AWS outposts.

Issue type (prefix)	Extra information	Recommended troubleshooting steps
EKS Runtime Monitoring enablement permission not granted	(may or may not show extra information)	1. If the extra information is available for this issue, fix the root cause and follow the next step. 2. Toggle EKS Runtime Monitoring to turn it off and then turn it on again. Ensure that the GuardDuty agent also gets deployed, whether automatically through GuardDuty or manually.
EKS Runtime Monitoring enablement resource provisioning in progress	(may or may not show extra information)	Not actionable. After you enable EKS Runtime Monitoring, the coverage status might remain `Unhealthy` until the resource provisioning step completes. The coverage status gets monitored and updated periodically.
Others (any other issue)	Error due to authorization failure	Toggle EKS Runtime Monitoring to turn it off and then turn it on again. Ensure that the GuardDuty agent also gets deployed, either automatically through GuardDuty or manually.

Troubleshooting steps for Addon creation/updation error with Addon issue code

Addon creation or updation error	Troubleshooting steps
EKS Addon Issue - `InsufficientNumberOfReplicas` : The add-on is unhealthy because it doesn't have the desired number of replicas.	• Using the issue message, you can identify and fix the root cause. You can start by describing your cluster. For example, use `kubectl describe pods` to identify the root cause for pod failure. After you fix the root cause, retry the step (add-on creation or update). • If the issue persists, validate that the VPC endpoint for your Amazon EKS cluster is correctly configured. For more information, see Validating VPC endpoint configuration.
EKS Addon Issue - `InsufficientNumberOfReplicas` : The add-on is unhealthy because one or more pods is not scheduled `0/x` nodes are available: `x Insufficient cpu. preemption: not eligible due to preemptionPolicy=Never` .	To resolve this issue, you can do one of the following: • Update pod priority of the GuardDuty agent: Configurable parameters and values by setting the `PriorityClass` to any one of the options that support the `preemptionPolicy` value as `PreemptLowerPriority` . For information about pod priority, see Pod Priority and Preemption in the *Kubernetes Documentation*. • Scale up the instance: For managing your resources and making optimal instance selection, see Manage compute resources by using nodes and Choose an optimal Amazon EC2 node instance type in the Amazon EKS User Guide.
EKS Addon Issue - `InsufficientNumberOfReplicas` : The add-on is unhealthy because one or more pods is not scheduled `0/x` nodes are available: `x Too many pods. preemption: not eligible due to preemptionPolicy=Never` .	
EKS Addon Issue - `InsufficientNumberOfReplicas` : The add-on is unhealthy because one or more pods is not scheduled `0/x` nodes are available: `1 Insufficient`	

Addon creation or updation error	Troubleshooting steps
`memory. preemption: not eligible due to preemptionPolicy=Never .`	**Note** The message shows `0/x` because GuardDuty reports only the first found error. The actual number of running pods in the GuardDuty daemonset might be greater than 0.

<table>
<tr><th>Addon creation or updation error</th><th>Troubleshooting steps</th></tr>
<tr><td>EKS Addon Issue - <code>InsufficientNumberOfReplicas</code> : The add-on is unhealthy because one or more pods have waiting containers <code>CrashLoopBackOff: Completed</code></td><td>You can view the logs associated with the pod and identify the issue. For information on how to do this, see Debug Running Pods in the Kubernetes Documentation.

Use the following checklist to troubleshoot this add-on issue:

• Validate that Runtime Monitoring is enabled.
• Validate that the Prerequisites for Amazon EKS cluster support, such as verified OS distributions and supported Kubernetes versions, are met.
• When you manage the security agent manually, confirm that you created a VPC endpoint for all the VPCs. When you enable GuardDuty automated configuration, you should still validate that the VPC endpoint gets created. For example, when using a shared VPC in automated configuration.

To validate this, see Validating VPC endpoint configuration.
• Confirm that the GuardDuty security agent is able to resolve the GuardDuty VPC endpoint private DNS. To know the endpoints, see Private DNS names for endpoints in Managing GuardDuty security agents.

To do this, you can use either <code>nslookup</code> tool on Windows or Mac, or <code>dig</code> tool on</td></tr>
</table>

<table>
<tr><th>Addon creation or updation error</th><th>Troubleshooting steps</th></tr>
<tr><td></td><td>Linux. When using nslookup, you can use the following command after replacing the Region <code>us-west-2</code> with your Region:
<pre>nslookup guardduty-data. us-west-2 .amazonaws.com</pre>
Validate that your GuardDuty VPC endpoint policy or the service control policy is not impacting <code>guardduty:SendSecurityTelemetry</code> action.</td></tr>
<tr><td>EKS Addon Issue - <code>InsufficientNumberOfReplicas</code> : The add-on is unhealthy because one or more pods have waiting containers <code>CrashLoopBackOff: Error</code></td><td>You can view the logs associated with the pod and identify the issue. For information on how to do this, see Debug Running Pods in the Kubernetes Documentation.

After you have identified the issue, use the following checklist to troubleshoot this:
Validate that Runtime Monitoring is enabled.
Validate that the Prerequisites for Amazon EKS cluster support, such as verified OS distributions and supported Kubernetes versions, are met.
The GuardDuty security agent is able to resolve the GuardDuty VPC endpoint private DNS. To know the endpoints, see Private DNS names for endpoints in Managing GuardDuty security agents.</td></tr>
</table>

Addon creation or updation error	Troubleshooting steps
EKS Addon Issue - `AdmissionRequestDenied` : admission webhook `"validate.kyverno.svc-fail"` denied the request: policy `DaemonSet/amazon-guardduty/aws-guardduty-agent` for resource violation: restrict-image-registries: `autogen-validate-registries` : ...	1. Amazon EKS cluster or the security administrator must review the security policy that is blocking the Addon update. 2. You must either disable the controller (`webhook`) or have the controller accept the requests from Amazon EKS.
EKS Addon Issue - `ConfigurationConflict` : Conflicts found when trying to apply. Will not continue due to resolve conflicts mode. `Conflicts: DaemonSet.apps aws-guardduty-agent - .spec.template.spec.containers[name="aws-guardduty-agent"].image`	When creating or updating the Addon, provide the `OVERWRITE` resolve conflict flag. This will potentially overwrite any changes that have been made directly to the related resources in Kubernetes by using the Kubernetes API. You can first Remove an Amazon EKS add-on from a cluster and then reinstall.

<table>
<tr><th>Addon creation or updation error</th><th>Troubleshooting steps</th></tr>
<tr><td>EKS Addon Issue - <code>AccessDenied: priorityclasses.scheduling.k8s.io "aws-guardduty-agent.priorityclass" is forbidden: User "eks:addon-manager" cannot patch resource "priorityclasses" in API group "scheduling.k8s.io" at the cluster scope</code></td>
<td rowspan="2">You must add the missing permission to the <code>eks:addon-cluster-admin ClusterRoleBinding</code> manually. Add the following yaml to <code>eks:addon-cluster-admin</code> :
<pre>
---
kind: ClusterRoleBinding
apiVersion: rbac.authorization
.k8s.io/v1
metadata:
  name: eks:addon-cluster-admin
subjects:
- kind: User
  name: eks:addon-manager
  apiGroup: rbac.authorization.k8s.io
roleRef:
  kind: ClusterRole
  name: cluster-admin
  apiGroup: rbac.authorization.k8s.io

</pre>
You can now apply this <code>yaml</code> to your Amazon EKS cluster by using the following command:
<pre>
kubectl apply -f eks-addon-cluster-
admin.yaml
</pre></td></tr>
<tr><td>AddonUpdationFailed: EKSAddonIssue - <code>AccessDenied: namespaces\"amazon-guardduty\"isforbidden:User\"eks:addon-manager\"cannotpatchresource\"namespaces\"inAPIgroup\"\"inthenamespace\"amazon-guardduty\"</code></td></tr>
<tr><td>EKS Addon Issue - <code>AccessDenied: admission webhook "validation.gatekeeper.sh" denied the request: [all-namespace-must-have-label-owner] All namespaces must have an `owner` label</code></td>
<td>You must either disable the controller or have the controller accept the requests from the Amazon EKS cluster.

Prior to creating or updating the add-on, you can also create a GuardDuty namespace and label it as <code>owner</code>.</td></tr>
</table>

Addon creation or updation error	Troubleshooting steps
EKS Addon Issue - `AccessDenied: admission webhook "validation.gatekeeper.sh" denied the request: [all-namespace-must-have-label-owner] All namespaces must have an `owner` label`	You must either disable the controller or have the controller accept the requests from the Amazon EKS cluster. Prior to creating or updating the add-on, you can also create a GuardDuty namespace and label it as `owner`.
EKS Addon Issue - `AccessDenied: admission webhook "validation.gatekeeper.sh" denied the request: [allowed-container-registries] container <aws-guardduty-agent> has an invalid image registry`	Add the image registry for GuardDuty to the `allowed-container-registries` in your admission controller. For more information, see *ECR repository for EKS v1.8.1-eks-build.2* in [Amazon ECR repository hosting GuardDuty agent]().

Setting up CPU and memory monitoring

After you enable Runtime Monitoring and assess that the coverage status of your cluster is **Healthy**, you can set up and view the insight metrics.

The following topics can help you evaluate how the deployed agent performs against the CPU and memory limits for the GuardDuty agent.

Setting up monitoring on Amazon ECS cluster

The following steps from the *Amazon CloudWatch User Guide* can help you evaluate how the deployed agent performs against the CPU and memory limits for the GuardDuty agent:

1. Setting up Container Insights on Amazon ECS for cluster- and service-level metrics
2. Amazon ECS Container Insights metrics

Setting up monitoring on Amazon EKS cluster

After the GuardDuty security agent gets deployed and you assess that the coverage status of your cluster is **Healthy**, you can set up and view the Container insight metrics.

Evaluate performance of the security agent

1. Setting up Container Insights on Amazon EKS and Kubernetes in the *Amazon CloudWatch User Guide*
2. Amazon EKS and Kubernetes Container Insights metrics in the *Amazon CloudWatch User Guide*

Manage performance with security agent v1.5.0 and above

With security agent v1.5.0 and above, when the insights indicate that the associated GuardDuty agent is reaching the assigned limits, you can configure specific parameters. For more information, see Configure EKS add-on parameters.

Using shared VPC with Runtime Monitoring

GuardDuty Runtime Monitoring supports using shared Amazon Virtual Private Cloud (Amazon VPC) for your AWS accounts that belong to the same organization in AWS Organizations. You can use shared VPC in two ways:

- **Automated agent configuration (Recommended)** – When GuardDuty automatically manages the security agent, it will also configure the Amazon VPC endpoint policy. This policy is based on your organization's shared VPC settings.

  You must enable automated agent configuration in the shared VPC owner account and all the participating accounts who will share this VPC.
- **Manually managed agent** – When you manually manage the security agent with shared VPC, you must update the VPC endpoint policy to allow corresponding accounts to access shared VPC. To do this, you can use the example policy shared in the following How it works section.

  For manual management scenarios involving participating accounts for shared VPC, the coverage status may not be accurate. To ensure up-to-date protection and coverage status of your resources, GuardDuty recommends enabling automated agent configuration for all the accounts that will use shared VPC.

Topics

- [How it works]
- [Prerequisites for using shared VPC]

How it works

The AWS accounts that belong to the same organization as the shared Amazon VPC owner account can also share the same Amazon VPC endpoint. Each of the accounts using the same Amazon VPC endpoint policy is called as the **participant AWS account** of the associated shared Amazon VPC.

The following example shows the default VPC endpoint policy of the shared VPC owner account and the participant account. The `aws:PrincipalOrgID` will show the organization ID associated with the shared VPC resource. The use of this policy is limited to the participant accounts present in the organization of the owner account.

Example Example shared VPC endpoint policy

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [{
            "Action": "*",
            "Resource": "*",
            "Effect": "Allow",
            "Principal": "*"
        },
        {
            "Condition": {
                "StringNotEquals": {
                    "aws:PrincipalOrgID": "o-abcdef0123"
                }
            },
            "Action": "*",
            "Resource": "*",
            "Effect": "Deny",
            "Principal": "*"
        }
    ]
}
```

With GuardDuty automatic agent configuration

When the owner account of the shared VPC enables Runtime Monitoring and automated agent configuration for any of the resources (Amazon EKS or AWS Fargate (Amazon ECS only)), all the shared VPCs become eligible for automatic installation of the shared Amazon VPC endpoint and the associated security group in the shared VPC owner account. GuardDuty retrieves the organization ID that is associated with the shared Amazon VPC.

GuardDuty creates an Amazon VPC endpoint when either the shared VPC owner account or the participating account needs it. Examples of needing an Amazon VPC endpoint include enabling GuardDuty, Runtime Monitoring, EKS Runtime Monitoring, or launching a new Amazon ECS-Fargate task. When these accounts enable Runtime Monitoring and automated agent configuration for any resource type, GuardDuty creates an Amazon VPC endpoint and sets the endpoint policy with the same organization ID as that of the shared VPC owner account. GuardDuty adds a `GuardDutyManaged` tag and sets it to `true` for the Amazon VPC endpoint that GuardDuty creates. If the shared Amazon VPC owner account has not enabled Runtime Monitoring or automated agent configuration for any of the resources, GuardDuty will not set the Amazon VPC endpoint policy. For information about configuring Runtime Monitoring and managing the security agent automatically in the shared VPC owner account, see Enabling GuardDuty Runtime Monitoring.

Using with manually managed agent

When you use shared VPC with manually managed agent, validate that there is no explicit `Deny` endpoint policy that blocks any account that needs to use the shared VPC. This will prevent the security agent from sending telemetry to GuardDuty, resulting in an `Unhealthy` coverage status. For setting up the endpoint policy, see Example shared VPC endpoint policy.

Runtime coverage may not be accurate in scenarios such as missing permissions to the shared VPC. You can continuously monitor resource coverage by following the steps for your resource type in Reviewing runtime coverage statistics and troubleshooting issues.

To ensure continuous Runtime Monitoring protection of your compute resources, GuardDuty recommends enabling automated agent configuration for the shared VPC owner account and all the participating accounts for your resources.

Prerequisites for using shared VPC

As a part of an initial setup, perform the following steps in the AWS account that you want to be the owner of the shared VPC:

1. **Creating an organization** – Create an organization by following the steps in Creating and managing an organization in the *AWS Organizations User Guide*.

   For information about adding or removing member accounts, see Managing AWS accounts in your organization.

2. **Creating a shared VPC resource** – You can create a shared VPC resource from the owner account. For more information, see Share your VPC subnets with other accounts in the *Amazon VPC User Guide*.

Prerequisites specific to GuardDuty Runtime Monitoring

The following list provides the prerequisites that are specific to GuardDuty:

- The owner account of the shared VPC and the participating account can be from different organizations in GuardDuty. However, they must belong to the same organization in AWS Organizations. This is required for GuardDuty to create an Amazon VPC endpoint and a security group for the shared VPC. For information about how shared VPCs work, see Share your VPC with other accounts in *Amazon VPC User Guide*.
- Enable Runtime Monitoring or EKS Runtime Monitoring, and GuardDuty automated agent configuration for any resource in the shared VPC owner account and the participant account. For more information, see Enabling Runtime Monitoring.

  If you have already completed these configurations, continue with the next step.
- When working with either an Amazon EKS or an Amazon ECS (AWS Fargate only) task, make sure to choose the shared VPC resource associated with the owner account and select its subnets.

Using Infrastructure as Code (IaC) with GuardDuty automated security agents

Use this section only if the following list applies to your use case:

- You use Infrastructure as Code (IaC) tools, such as AWS Cloud Development Kit (AWS CDK) and Terraform, to manage your AWS resources, and
- You need to enable GuardDuty automated agent configuration for one or more resource types - Amazon EKS, Amazon EC2, or Amazon ECS-Fargate.

IaC resource dependency graph overview

When you enable GuardDuty automated agent configuration for a resource type, GuardDuty automatically creates a VPC endpoint and a security group associated with this VPC endpoint, and installs the security agent for this resource type. By default, GuardDuty will delete the VPC endpoint and the associated security group only after you disable Runtime Monitoring. For more information, see Disabling, uninstalling, and cleaning up resources in Runtime Monitoring.

When you use an IaC tool, it maintains a dependency graph of resources. At the time of deletion of resources using the IaC tool, it only deletes resources that can be tracked as a part of dependency graph of resources. IaC tools may not know about the resources that are created outside of their specified configuration. For example, you create a VPC with an IaC tool and then add a security group to this VPC by using AWS console or an API operation. In the resource dependency graph, the VPC resource that you create depends on the associated security group. If you delete this VPC resource by using the IaC tool, then you will get an error. The way to get around this error is to delete the associated security group manually or to update the IaC configuration to include this added resource.

Common issue - Deleting resources in IaC

When using GuardDuty automated agent configuration, you may want to delete a resource (Amazon EKS, Amazon EC2, or Amazon ECS-Fargate) that you created by using an IaC tool. However, this resource is dependent on a VPC endpoint that GuardDuty created. This prevents the IaC tool to delete the resource by itself and requires you to disable Runtime Monitoring, that further deletes the VPC endpoint automatically.

For example, when you attempt to delete the VPC endpoint that GuardDuty created on your behalf, you will get an error similar to the following examples.

Example

Error example when using CDK

```
The following resource(s) failed to delete:
 [mycdkvpcapplicationpublicsubnet1Subnet1SubnetEXAMPLE1, mycdkvpcapplicationprivatesubnet1Subne
Resource handler returned message: "The subnet 'subnet-APKAEIVFHP46CEXAMPLE' has
 dependencies and cannot be deleted. (Service: Ec2, Status Code: 400, Request
 ID: e071c3c5-7442-4489-838c-0dfc6EXAMPLE)" (RequestToken: 4381cff8-6240-208a-8357-5557b7EXAMPL
 HandlerErrorCode: InvalidRequest)
```

Example

Error example when using Terraform

```
module.vpc.aws_subnet.private[1]: Still destroying... [id=subnet-APKAEIVFHP46CEXAMPLE,
 19m50s elapsed]
module.vpc.aws_subnet.private[1]: Still destroying... [id=subnet-APKAEIVFHP46CEXAMPLE,
 20m0s elapsed]

Error: deleting EC2 Subnet (subnet-APKAEIBAERJR2EXAMPLE): DependencyViolation: The
 subnet 'subnet-APKAEIBAERJR2EXAMPLE' has dependencies and cannot be deleted.
       status code: 400, request id: e071c3c5-7442-4489-838c-0dfc6EXAMPLE
```

Solution - Prevent resource deletion issue

This section helps you manage the VPC endpoint and security group independent of GuardDuty.

To gain complete ownership of the resources configured by using the IaC tool, perform the following steps in the listed order:

1. Create a VPC. To allow ingress permission, associate a GuardDuty VPC endpoint with the security group, to this VPC.
2. Enable GuardDuty automated agent configuration for your resource type

After you complete the preceding steps, GuardDuty will not create its own VPC endpoint and will reuse the one that you created by using the IaC tool.

For information about creating your own VPC, see Create a VPC only in the *Amazon VPC Transit Gateways*. For information about creating a VPC endpoint, see the following section for your resource type:

- For Amazon EC2, see Prerequisite – Creating Amazon VPC endpoint manually.
- For Amazon EKS, see Prerequisite – Creating an Amazon VPC endpoint.

Collected runtime event types that GuardDuty uses

The GuardDuty security agent collects the following events types and sends them to the GuardDuty backend for threat detection and analysis. GuardDuty doesn't make these events accessible to you. If GuardDuty detects a potential threat and generates a Runtime Monitoring finding types, you can view the corresponding finding details.

For information about how GuardDuty uses the collected event types in Runtime Monitoring, see Opting out of using your data for service improvement.

Process events

Process events represent information associated with the processes running on Amazon EC2 instances and container workloads. The following table includes the field names and descriptions of the process events that Runtime Monitoring collects to detect potential threats.

Field name	Description
Process name	Name of the observed process.
Process Path	Absolute path of the process executable.
Process ID	The ID assigned to the process by the operating system.
Namespace PID	The process ID of the process in a secondary PID namespace other than the host level PID namespace. For processes inside a container, it is the process ID observed inside the container.
Process User ID	The unique ID of the user that executed the process.
Process UUID	The unique ID assigned to the process by GuardDuty.
Process GID	Process ID of the process group.
Process EGID	Effective group ID of the process group.
Process EUID	Effective user ID of the process.
Process User Name	The user name that executed the process.

Field name	Description
Process Start Time	The time when the process was created. This field is in the UTC date string format (`2023-03-22T19:37:20.168Z`).
Process Executable SHA-256	The `SHA256` hash of the process executable.
Process Script Path	Path of the script file that was executed.
Process Environment Variable	The environment variable made available to the process. Only `LD_PRELOAD` and `LD_LIBRARY_PATH` get collected.
Process Present Working Directory (PWD)	Present working directory of the process.
Parent process	Process details of the parent process. A parent process is a process that created the observed process.
Command Line Arguments Presently, this field is limited to specific agent versions corresponding to the resource type: • Fargate (Amazon ECS only) with GuardDuty security agent v1.0.0 and above. • Amazon EC2 instances with GuardDuty security agent v1.0.0 and above. • Amazon EKS clusters with security agent v1.4.0 and above. For more information, see [GuardDuty security agent release versions].	Command-line arguments provided at the time of process execution. This field might contain sensitive customer data.

Container events

Container events represent information associated with activities of the container workloads. The following table includes the field names and descriptions of the container workload events that Runtime Monitoring collects to detect potential threats.

Field name	Description
Container Name	Name of the container. When available, this field displays the value of the label `io.kubenetes.container.name` .
Container UID	The unique ID of the container assigned by the container runtime.
Container Runtime	The container runtime (such as `docker` or `containerd`) used to run the container.
Container Image ID	The ID of the container image.
Container Image Name	Name of the container image.

AWS Fargate (Amazon ECS only) task events

Fargate-Amazon ECS task events represent activities associated with Amazon ECS tasks running on Fargate computes. The following table includes the field names and descriptions of the Amazon ECS-Fargate task events that Runtime Monitoring collects to detect potential threats.

Field name	Description
Task Amazon Resource Name (ARN)	The ARN of the task.
Cluster Name	The name of the Amazon ECS cluster.
Family Name	The task definition's family name. The `family` is used as a name for the task definition that is used to launch the task.

Field name	Description
Service Name	The name of the Amazon ECS service, if the task was launched as part of a service.
Launch Type	The infrastructure on which your task runs. For Runtime Monitoring with resource type as `ECSCluster`, the launch type could be either `EC2` or `FARGATE`.
CPU	The number of CPU units used by the task as expressed in the task definition.

Kubernetes pod events

The following table includes the field names and descriptions of the Kubernetes pod events that Runtime Monitoring collects to detect potential threats.

Field name	Description
Pod ID	The ID of the Kubernetes pod.
Pod name	Name of the Kubernetes pod.
Pod Namespace	Name of the Kubernetes namespace to which the Kubernetes workload belongs.
Kubernetes Cluster Name	Name of the Kubernetes cluster.

Domain Name System (DNS) events

The Domain Name System (DNS) events includes details of the DNS queries made by your resource types and corresponding responses. The following table includes the field names and descriptions of the DNS events that Runtime Monitoring collects to detect potential threats.

Field name	Description
Socket Type	Type of socket to indicate communication semantics. For example, SOCK_RAW.
Address Family	Represents the communication protocol associated with the address. For example, the address family AF_INET is used for IP v4 protocol.
Direction ID	The ID of the connection direction.
Protocol Number	The layer 4 protocol number such as 17 for UDP and 6 for TCP.
DNS Remote Endpoint IP	The remote IP of the connection.
DNS Remote Endpoint Port	The port number of the connection.
DNS Local Endpoint IP	The local IP of the connection.
DNS Local Endpoint Port	The port number of the connection.
DNS Payload	The payload of DNS packets that contains DNS queries and responses.

Open events

Open events are associated with file access and modification. The following table includes the field names and descriptions of the open events that Runtime Monitoring collects to detect potential threats.

Field name	Description
Filepath	Path of the file that is opened in this event.
Flags	Describes the file access mode, such as read-only, write-only, and read-write.

Load module event

The following table includes the field name and description of the load module event that Runtime Monitoring collects to detect potential threats.

Field name	Description
Module Name	Name of the module loaded into the kernel.

Mprotect events

Mprotect events provide information about changes to the memory protection settings of the processes running on the monitored systems. The following table includes the field names and descriptions of the Mprotect events that Runtime Monitoring collects to detect potential threats.

Field name	Description
Address Range	The address range for which the access protections were modified.
Memory Regions	Specifies the Region of a process's address space such as stack and heap.
Flags	Represents options that control the behavior of this event.

Mount events

Mount events provide information associated with the mounting and unmounting of file systems on your monitored resource. The following table includes the field names and descriptions of the mount events that Runtime Monitoring collects to detect potential threats.

Field name	Description
Mount Target	The path where the mount source is mounted.
Mount Source	The path on the host that is mounted at the mount target.

Field name	Description
Filesystem Type	Represents the type of mounted fileSystem.
Flags	Represents options that control the behavior of this event.

Link events

Link events provide visibility into the file system link management activities in your monitored resources. The following table includes the field names and descriptions of the link events that Runtime Monitoring collects to detect potential threats.

Field name	Description
Link Path	Path where the hard link gets created.
Target Path	Path of the file at which the hard link points.

Symlink events

Symlink events provide visibility into the file system symbolic link management activities in your monitored resources. The following table includes the field names and descriptions of the symlink events that Runtime Monitoring collects to detect potential threats.

Field name	Description
Link Path	Path where the symbolic link is created.
Target Path	Path of the file at which the symbolic link points.

Dup events

Dup events provide visibility into the duplication of file descriptors by processes running on the monitored resources. The following table includes the field names and descriptions of the dup events that Runtime Monitoring collects to detect potential threats.

Field name	Description
Old File Descriptor	A file descriptor that represents an open file object.
New File Descriptor	A new file descriptor that is a duplicate of the old file descripto r. Both the old and new file descriptors represent the same open file object.
Dup Remote Endpoint IP	The remote IP address of the network socket represented by the old file descriptor. Only applicable when the old file descriptor represents a network socket.
Dup Remote Endpoint Port	The remote port of the network socket represented by the old file descriptor. Only applicable when the old file descriptor represents a network socket.
Dup Local Endpoint IP	The local IP address of the network socket represented by the old file descriptor. Only applicable when the old file descriptor represents a network socket.
Dup Local Endpoint Port	The local port of the network socket represented by the old file descriptor. Only applicable when the old file descriptor represents a network socket.

Memory map event

The following table includes the field name and description of the memory map events that Runtime Monitoring collects to detect potential threats.

Field name	Description
Filepath	Path of the file to which the memory is mapped.

Socket events

Socket events provide information about the network socket connections used in the activities of the monitored resources. The following table includes the field names and descriptions of the socket events that Runtime Monitoring collects to detect potential threats.

Field name	Description
Address family	Represents the communication protocol associated with the address. For example, the address family `AF_INET` is used for IP version of 4 protocol.
Socket Type	Type of socket to indicate communication semantics. For example, `SOCK_RAW`.
Protocol number	Specifies a particular protocol within the address family. Usually there is a single protocol in address families. For example, the address family `AF_INET` only has the IP protocol.

Connect events

Connect events provide visibility into the network connections established by the processes on your monitored resources. The following table includes the field names and descriptions of the connect events that Runtime Monitoring collects to detect potential threats.

Field name	Description
Address family	Represents the communication protocol associated with the address. For example, the address family `AF_INET` is used for IP v4 protocol.
Socket Type	Type of socket to indicate communication semantics. For example, `SOCK_RAW`.
Protocol Number	Specifies a particular protocol within the address family. Usually there is a single protocol in address families. For example, the address family `AF_INET` only has the IP protocol.

Field name	Description
Filepath	Path of the socket file if the address family is `AF_UNIX`.
Remote Endpoint IP	The remote IP of the connection.
Remote Endpoint Port	The port number of the connection.
Local Endpoint IP	The local IP of the connection.
Local Endpoint Port	The port number of the connection.

Process VM Readv events

Process VM readv events provide visibility into the read operations performed by the processes on their own virtual memory regions. The following table includes the field names and descriptions of the process VM readv events that Runtime Monitoring collects to detect potential threats.

Field name	Description
Flags	Represents options that control the behavior of this event.
Target PID	Process ID of the process from which memory is being read.
Target Process UUID	The unique ID of the target process.
Target Executable Path	The absolute path of the target process executable file.

Process VM Writev events

Process VM writev events provide visibility into the write operations performed by the processes on their own virtual memory regions. The following table includes the field names and descriptions of the process VM writev events that Runtime Monitoring collects to detect potential threats.

Field name	Description
Flags	Represents options that control the behavior of this event.

Field name	Description
Target PID	Process ID of the process to which memory is being written.
Target Process UUID	The unique ID of the target process.
Target Executable Path	The absolute path of the target process executable file.

Process trace (Ptrace) events

Process trace (Ptrace) system call is a debugging and tracing mechanism that allows one process (tracer) to observe and control the execution of another process (tracee). This provides the tracer with the ability to inspect and modify the target process's memory, registers, and execution flow.

Ptrace events provide visibility into the use of ptrace system call by processes running on the monitored resources. The following table includes the field names and descriptions of the ptrace events that Runtime Monitoring collects to detect potential threats.

Field name	Description
Target PID	Process ID of the target process.
Target Process UUID	The unique ID of the target process.
Target Executable Path	The absolute path of the target process executable file.
Flags	Represents options that control the behavior of this event.

Bind events

Bind events provide visibility into binding of network sockets by processes running on the monitored resources. The following table includes the field names and descriptions of the bind events that Runtime Monitoring collects to detect potential threats.

Field name	Description
Address Family	Represents the communication protocol associated with the address. For example, the address family `AF_INET` is used for IP v4 protocol.
Socket type	Type of socket to indicate communication semantics. For example, `SOCK_RAW`.
Protocol number	The layer 4 protocol number such as 17 for UDP and 6 for TCP.
Local endpoint IP	The local IP of the connection.
Local endpoint port	The port number of the connection.

Listen events

Listen events provide visibility into the listening state of network sockets, indicating whether or not a network socket is ready to accept incoming connections. A process running on your monitored resource sets the network socket to a listening state. The following table includes the field names and descriptions of the listen events that Runtime Monitoring collects to detect potential threats.

Field name	Description
Address Family	Represents the communication protocol associated with the address. For example, the address family `AF_INET` is used for IP v4 protocol.
Socket type	Type of socket to indicate communication semantics. For example, `SOCK_RAW`.
Protocol number	The layer 4 protocol number such as 17 for UDP and 6 for TCP.
Local endpoint IP	The local IP of the connection.
Local endpoint port	The port number of the connection.

Rename events

Rename events provide information about the renaming of files and directories by processes running on the monitored resources. The following table includes the field names and descriptions of the rename events that Runtime Monitoring collects to detect potential threats.

Field name	Description
Filepath	Path where the file that is renamed.
Target	The new path of the file.

Set user ID (UID) events

Set user ID (UID) events provide visibility into the changes made to the user ID (UID) associated with the running processes on your monitored resources. The following table includes the field names and descriptions of the set UID events that Runtime Monitoring collects to detect potential threats.

Field name	Description
New EUID	The new effective user ID of the process.
New UID	The new user ID of the process.

Chmod events

Chmod events provide visibility into the changes in the permissions (mode) of files and directories on the monitored resources. The following table includes the field names and descriptions of the chmod events that Runtime Monitoring collects to detect potential threats.

Field name	Description
Filepath	Path of the file that invokes this event.
Filemode	The updated access permissions for the associated file.

Amazon ECR repository hosting GuardDuty agent

The following sections list the Amazon Elastic Container Registry (Amazon ECR) repositories where GuardDuty hosts the security agent that gets deployed on your Amazon EKS and Amazon ECS clusters.

The prerequisite to Prerequisites for container image access requires you to provide a task execution role that has certain Amazon Elastic Container Registry (Amazon ECR) permissions. To further restrict these permissions, you can add the Amazon ECR repository URI that hosts the GuardDuty agent for Fargate-Amazon ECS resources.

ECR repository for EKS agent versions 1.11.0 - 1.8.1 (eks.build.2)

When you enable GuardDuty automated configuration for Runtime Monitoring for EKS, GuardDuty will deploy this agent version to your Amazon EKS clusters. For information about enabling automated agent, see Managing security agent automatically for Amazon EKS resources.

The following table shows the Amazon ECR repository URIs where the GuardDuty security agent versions `1.11.0.eks.build.2`, `1.10.0.eks.build.2`, `1.9.0.eks.build.2`, and `1.8.0.eks.build.2` for Amazon EKS are hosted.

AWS Region	Amazon ECR repository URI
US West (Oregon)	`602401143452.dkr.ecr.us-west-2.amazonaws.com`
	`039403964562.dkr.ecr.us-west-2.amazonaws.com`
Europe (Paris)	`602401143452.dkr.ecr.eu-west-3.amazonaws.com`
	`113643092156.dkr.ecr.eu-west-3.amazonaws.com`
Asia Pacific (Mumbai)	`602401143452.dkr.ecr.ap-south-1.amazonaws.com`

AWS Region	Amazon ECR repository URI
	610108029387.dkr.ecr.ap-south-1.amazonaws.com
Asia Pacific (Hyderabad)	900889452093.dkr.ecr.ap-south-2.amazonaws.com
	618745550137.dkr.ecr.ap-south-2.amazonaws.com
Canada (Central)	602401143452.dkr.ecr.ca-central-1.amazonaws.com
	001188825231.dkr.ecr.ca-central-1.amazonaws.com
Canada West (Calgary)	761377655185.dkr.ecr.ca-west-1.amazonaws.com
	–
Middle East (UAE)	759879836304.dkr.ecr.me-central-1.amazonaws.com
	601769779514.dkr.ecr.me-central-1.amazonaws.com
Europe (London)	602401143452.dkr.ecr.eu-west-2.amazonaws.com
	109118265657.dkr.ecr.eu-west-2.amazonaws.com
US West (N. California)	602401143452.dkr.ecr.us-west-1.amazonaws.com
	373421517865.dkr.ecr.us-west-1.amazonaws.com

AWS Region	Amazon ECR repository URI
US East (N. Virginia)	602401143452.dkr.ecr.us-east-1.amazonaws.com
	031903291036.dkr.ecr.us-east-1.amazonaws.com
US East (Ohio)	602401143452.dkr.ecr.us-east-2.amazonaws.com
	591382732059.dkr.ecr.us-east-2.amazonaws.com
Europe (Ireland)	602401143452.dkr.ecr.eu-west-1.amazonaws.com
	673884943994.dkr.ecr.eu-west-1.amazonaws.com
South America (São Paulo)	602401143452.dkr.ecr.sa-east-1.amazonaws.com
	941219317354.dkr.ecr.sa-east-1.amazonaws.com
Europe (Stockholm)	602401143452.dkr.ecr.eu-north-1.amazonaws.com
	366771026645.dkr.ecr.eu-north-1.amazonaws.com
Europe (Frankfurt)	602401143452.dkr.ecr.eu-central-1.amazonaws.com
	409493279830.dkr.ecr.eu-central-1.amazonaws.com

AWS Region	Amazon ECR repository URI
Europe (Zurich)	900612956339.dkr.ecr.eu-central-2.amazonaws.com
	718440343717.dkr.ecr.eu-central-2.amazonaws.com
Asia Pacific (Singapore)	602401143452.dkr.ecr.ap-southeast-1.amazonaws.com
	584580519942.dkr.ecr.ap-southeast-1.amazonaws.com
Asia Pacific (Sydney)	602401143452.dkr.ecr.ap-southeast-2.amazonaws.com
	011662287384.dkr.ecr.ap-southeast-2.amazonaws.com
Asia Pacific (Jakarta)	296578399912.dkr.ecr.ap-southeast-3.amazonaws.com
	617474730032.dkr.ecr.ap-southeast-3.amazonaws.com
Asia Pacific (Tokyo)	602401143452.dkr.ecr.ap-northeast-1.amazonaws.com
	781592569369.dkr.ecr.ap-northeast-1.amazonaws.com
Asia Pacific (Seoul)	602401143452.dkr.ecr.ap-northeast-2.amazonaws.com
	732248494576.dkr.ecr.ap-northeast-2.amazonaws.com

AWS Region	Amazon ECR repository URI
Asia Pacific (Osaka)	602401143452.dkr.ecr.ap-northeast-3.amazonaws.com
	810724417379.dkr.ecr.ap-northeast-3.amazonaws.com
Asia Pacific (Hong Kong)	800184023465.dkr.ecr.ap-east-1.amazonaws.com
	790429075973.dkr.ecr.ap-east-1.amazonaws.com
Middle East (Bahrain)	558608220178.dkr.ecr.me-south-1.amazonaws.com
	541829937850.dkr.ecr.me-south-1.amazonaws.com
Europe (Milan)	590381155156.dkr.ecr.eu-south-1.amazonaws.com
	528450769569.dkr.ecr.eu-south-1.amazonaws.com
Europe (Spain)	455263428931.dkr.ecr.eu-south-2.amazonaws.com
	531047660167.dkr.ecr.eu-south-2.amazonaws.com
Africa (Cape Town)	877085696533.dkr.ecr.af-south-1.amazonaws.com
	379032919888.dkr.ecr.af-south-1.amazonaws.com

AWS Region	Amazon ECR repository URI
Asia Pacific (Melbourne)	491585149902.dkr.ecr.ap-southeast-4.amazonaws.com
	750462861327.dkr.ecr.ap-southeast-4.amazonaws.com
Israel (Tel Aviv)	066635153087.dkr.ecr.il-central-1.amazonaws.com
	292660727137.dkr.ecr.il-central-1.amazonaws.com
Asia Pacific (Malaysia)	151610086707.dkr.ecr.ap-southeast-5.amazonaws.com
Asia Pacific (Thailand)	121268973566.dkr.ecr.ap-southeast-7.amazonaws.com
Mexico (Central)	730335286997.dkr.ecr.mx-central-1.amazonaws.com
Asia Pacific (Taipei)	533267051163.dkr.ecr.ap-east-2.amazonaws.com

ECR repository for EKS agent version 1.8.1 (`eks.build.1`)

This section provides the Amazon ECR repository for the Amazon EKS agent version **1.8.1 (v1.8.1-eks-build.1)**. If you're using v1.8.1-eks-build.1, GuardDuty recommends switching to the default agent version which is usually the latest agent version. To do so, identify the latest agent from Released agent versions for Amazon EKS resources, and then perform the steps in Updating security agent manually for Amazon EKS resources.

The following table shows the Amazon ECR repository URIs where GuardDuty security agent version `1.8.1-eks-build.1` for Amazon EKS is hosted.

AWS Region	Amazon ECR repository URI
US West (Oregon)	`039403964562.dkr.ecr.us-west-2.amazonaws.com`
Europe (Paris)	`113643092156.dkr.ecr.eu-west-3.amazonaws.com`
Asia Pacific (Mumbai)	`610108029387.dkr.ecr.ap-south-1.amazonaws.com`
Asia Pacific (Hyderabad)	`618745550137.dkr.ecr.ap-south-2.amazonaws.com`
Canada (Central)	`001188825231.dkr.ecr.ca-central-1.amazonaws.com`
Middle East (UAE)	`601769779514.dkr.ecr.me-central-1.amazonaws.com`
Europe (London)	`109118265657.dkr.ecr.eu-west-2.amazonaws.com`
US West (N. California)	`373421517865.dkr.ecr.us-west-1.amazonaws.com`
US East (N. Virginia)	`031903291036.dkr.ecr.us-east-1.amazonaws.com`
US East (Ohio)	`591382732059.dkr.ecr.us-east-2.amazonaws.com`
Europe (Ireland)	`673884943994.dkr.ecr.eu-west-1.amazonaws.com`
South America (São Paulo)	`941219317354.dkr.ecr.sa-east-1.amazonaws.com`

AWS Region	Amazon ECR repository URI
Europe (Stockholm)	366771026645.dkr.ecr.eu-north-1.amazonaws.com
Europe (Frankfurt)	409493279830.dkr.ecr.eu-central-1.amazonaws.com
Europe (Zurich)	718440343717.dkr.ecr.eu-central-2.amazonaws.com
Asia Pacific (Singapore)	584580519942.dkr.ecr.ap-southeast-1.amazonaws.com
Asia Pacific (Sydney)	011662287384.dkr.ecr.ap-southeast-2.amazonaws.com
Asia Pacific (Jakarta)	617474730032.dkr.ecr.ap-southeast-3.amazonaws.com
Asia Pacific (Tokyo)	781592569369.dkr.ecr.ap-northeast-1.amazonaws.com
Asia Pacific (Seoul)	732248494576.dkr.ecr.ap-northeast-2.amazonaws.com
Asia Pacific (Osaka)	810724417379.dkr.ecr.ap-northeast-3.amazonaws.com
Asia Pacific (Hong Kong)	790429075973.dkr.ecr.ap-east-1.amazonaws.com
Middle East (Bahrain)	541829937850.dkr.ecr.me-south-1.amazonaws.com
Europe (Milan)	528450769569.dkr.ecr.eu-south-1.amazonaws.com

AWS Region	Amazon ECR repository URI
Europe (Spain)	531047660167.dkr.ecr.eu-south-2.amazonaws.com
Africa (Cape Town)	379032919888.dkr.ecr.af-south-1.amazonaws.com
Asia Pacific (Melbourne)	750462861327.dkr.ecr.ap-southeast-4.amazonaws.com
Israel (Tel Aviv)	292660727137.dkr.ecr.il-central-1.amazonaws.com

ECR Repository for GuardDuty agent on AWS Fargate (Amazon ECS only)

As a prerequisite to using Runtime Monitoring for Amazon ECS-Fargate, you must Prerequisites for container image access. The GuardDuty agent sidecar container image is stored in Amazon ECR, with its image layers stored in Amazon S3. For more information, see How Runtime Monitoring works with Fargate (Amazon ECS only).

The following table shows the Amazon ECR repositories that hosts the GuardDuty agent for AWS Fargate (Amazon ECS only) for each AWS Region.

AWS Region	Amazon ECR repository URI
US West (Oregon)	733349766148.dkr.ecr.us-west-2.amazonaws.com/aws-guardduty-agent-fargate
Europe (Paris)	665651866788.dkr.ecr.eu-west-3.amazonaws.com/aws-guardduty-agent-fargate
Asia Pacific (Mumbai)	251508486986.dkr.ecr.ap-south-1.amazonaws.com/aws-guardduty-agent-fargate

AWS Region	Amazon ECR repository URI
Asia Pacific (Hyderabad)	950823858135.dkr.ecr.ap-south-2.amazonaws.com/aws-guardduty-agent-fargate
Canada (Central)	354763396469.dkr.ecr.ca-central-1.amazonaws.com/aws-guardduty-agent-fargate
Middle East (UAE)	000014521398.dkr.ecr.me-central-1.amazonaws.com/aws-guardduty-agent-fargate
Europe (London)	892757235363.dkr.ecr.eu-west-2.amazonaws.com/aws-guardduty-agent-fargate
US West (N. California)	684579721401.dkr.ecr.us-west-1.amazonaws.com/aws-guardduty-agent-fargate
US East (N. Virginia)	593207742271.dkr.ecr.us-east-1.amazonaws.com/aws-guardduty-agent-fargate
US East (Ohio)	307168627858.dkr.ecr.us-east-2.amazonaws.com/aws-guardduty-agent-fargate
Europe (Ireland)	694911143906.dkr.ecr.eu-west-1.amazonaws.com/aws-guardduty-agent-fargate
South America (São Paulo)	758426053663.dkr.ecr.sa-east-1.amazonaws.com/aws-guardduty-agent-fargate

AWS Region	Amazon ECR repository URI
Europe (Stockholm)	591436053604.dkr.ecr.eu-north-1.amazonaws.com/aws-guardduty-agent-fargate
Europe (Frankfurt)	323658145986.dkr.ecr.eu-central-1.amazonaws.com/aws-guardduty-agent-fargate
Europe (Zurich)	529164026651.dkr.ecr.eu-central-2.amazonaws.com/aws-guardduty-agent-fargate
Asia Pacific (Singapore)	174946120834.dkr.ecr.ap-southeast-1.amazonaws.com/aws-guardduty-agent-fargate
Asia Pacific (Sydney)	005257825471.dkr.ecr.ap-southeast-2.amazonaws.com/aws-guardduty-agent-fargate
Asia Pacific (Jakarta)	510637619217.dkr.ecr.ap-southeast-3.amazonaws.com/aws-guardduty-agent-fargate
Asia Pacific (Tokyo)	533107202818.dkr.ecr.ap-northeast-1.amazonaws.com/aws-guardduty-agent-fargate
Asia Pacific (Seoul)	914738172881.dkr.ecr.ap-northeast-2.amazonaws.com/aws-guardduty-agent-fargate
Asia Pacific (Osaka)	273192626886.dkr.ecr.ap-northeast-3.amazonaws.com/aws-guardduty-agent-fargate

AWS Region	Amazon ECR repository URI
Asia Pacific (Hong Kong)	258348409381.dkr.ecr.ap-east-1.amazonaws.com/aws-guardduty-agent-fargate
Middle East (Bahrain)	536382113932.dkr.ecr.me-south-1.amazonaws.com/aws-guardduty-agent-fargate
Europe (Milan)	266869475730.dkr.ecr.eu-south-1.amazonaws.com/aws-guardduty-agent-fargate
Europe (Spain)	919611009337.dkr.ecr.eu-south-2.amazonaws.com/aws-guardduty-agent-fargate
Africa (Cape Town)	197869348890.dkr.ecr.af-south-1.amazonaws.com/aws-guardduty-agent-fargate
Asia Pacific (Melbourne)	251357961535.dkr.ecr.ap-southeast-4.amazonaws.com/aws-guardduty-agent-fargate
Israel (Tel Aviv)	870907303882.dkr.ecr.il-central-1.amazonaws.com/aws-guardduty-agent-fargate
Asia Pacific (Malaysia)	156041399949.dkr.ecr.ap-southeast-5.amazonaws.com/aws-guardduty-agent-fargate
Asia Pacific (Thailand)	054037130133.dkr.ecr.ap-southeast-7.amazonaws.com/aws-guardduty-agent-fargate

AWS Region	Amazon ECR repository URI
Canada West (Calgary)	`339712888787.dkr.ecr.ca-west-1.amazonaws.com/aws-guardduty-agent-fargate`
Mexico (Central)	`311141559934.dkr.ecr.mx-central-1.amazonaws.com/aws-guardduty-agent-fargate`
Asia Pacific (Taipei)	`259886477082.dkr.ecr.ap-east-2.amazonaws.com/aws-guardduty-agent-fargate`

Two security agents on same underlying host

Amazon EC2 instances can support multiple types of workloads. When you configure automated security agent on an Amazon EC2 instance, the same EC2 instance might have another security agent through EKS.

Overview

Consider a scenario where you have enabled Runtime Monitoring. Now, you enable the automated agent for Amazon EKS through GuardDuty. You have also enabled the automated agent for Amazon EC2. It may happen that the same underlying host gets installed with two security agents - one for Amazon EKS and the other for Amazon EC2. This could result in two security agents running inside the same host, collecting runtime events and sending them to GuardDuty, and potentially generating duplicate findings.

Impact

- When there is more than one security agent running on the same host, your account may experience double the amount of CPU and memory processing needs. For information about the CPU and memory limits for each resource type, see Prerequisites for that resource.
- GuardDuty has designed the Runtime Monitoring feature in a way that even if there is an overlap of two security agents collecting runtime events from the same underlying host, your account will only be charged for one stream of runtime events.

How GuardDuty handles multiple agents

GuardDuty detects when two security agents are running on the same host and designates only one of them to be the security agent that actively collects runtime events. The second agent will consume minimum system resources so as to prevent any impact to the performance of your applications.

GuardDuty considers the following scenarios:

- When an EC2 instance falls under the scope of both Amazon EKS and Amazon EC2 security agents, the EKS security agent takes priority. This will apply only when you use the security agent v1.1.0 or above for Amazon EC2. Older agent versions will continue to run and collect runtime events because older agent versions are not affected by prioritization.
- When both Amazon EKS and Amazon EC2 have GuardDuty managed security agents and your Amazon EC2 instance is also SSM managed, both the security agents will be installed at the host level. Once the agents are installed, GuardDuty decides which security agent will keep running. When both the security agents are running, eventually only one of them will collect runtime events.
- When the security agents associated with both EC2 and EKS run at the same time, GuardDuty might generate duplicate findings during the overlap period only.

  This can happen when:

  - Security agents for both EC2 and EKS are configured through GuardDuty (automatically), or
  - Your Amazon EKS resource has automated security agent.
- When the EKS security agent is already running, if you deploy the EC2 security agent manually on the same underlying host and meet all the prerequisites, GuardDuty might not install a second security agent.

EKS Runtime Monitoring in GuardDuty

EKS Runtime Monitoring provides runtime threat detection coverage for Amazon Elastic Kubernetes Service (Amazon EKS) nodes and containers within your AWS environment. EKS Runtime Monitoring uses a GuardDuty security agent that adds runtime visibility into individual EKS workloads, for example, file access, process execution, and network connections. The GuardDuty security agent helps GuardDuty identify specific containers within your EKS clusters that are potentially compromised. It can also detect attempts to escalate privileges from an individual container to the underlying EC2 host, and the broader AWS environment.

With the availability of Runtime Monitoring, GuardDuty has consolidated the console experience for EKS Runtime Monitoring into Runtime Monitoring. GuardDuty will not migrate your EKS Runtime Monitoring settings on your behalf automatically. This requires an action at your end. If you want to continue using only EKS Runtime Monitoring, you can use the APIs or AWS CLI to check and update the existing configuration status for EKS Runtime Monitoring. However, GuardDuty recommends Migrating from EKS Runtime Monitoring to Runtime Monitoring and using Runtime Monitoring to monitor your Amazon EKS clusters.

Topics

- Configuring EKS Runtime Monitoring for multiple-account environments (API)
- Configuring EKS Runtime Monitoring for a standalone account (API)
- Migrating from EKS Runtime Monitoring to Runtime Monitoring

Configuring EKS Runtime Monitoring for multiple-account environments (API)

In a multiple-account environments, only the delegated GuardDuty administrator account can enable or disable EKS Runtime Monitoring for the member accounts, and manage GuardDuty agent management for the EKS clusters belonging to the member accounts in their organization. The GuardDuty member accounts can't modify this configuration from their accounts. The delegated GuardDuty administrator account account manages their member accounts using AWS Organizations. For more information about multi-account environments, see Managing multiple accounts.

Configuring EKS Runtime Monitoring for delegated GuardDuty administrator account

This section provides steps to configure EKS Runtime Monitoring and manage the GuardDuty security agent for the EKS clusters that belong to the delegated GuardDuty administrator account.

Based on the Approaches to manage GuardDuty security agent in Amazon EKS clusters, you can choose a preferred approach and follow the steps as mentioned in the following table.

Preferred approach to manage GuardDuty security agent	Steps
Manage security agent through GuardDuty (Monitor all EKS clusters)	Run the updateDetector API by using your own regional detector ID and passing the `features` object name as `EKS_RUNTIME_MONITORING` and status as `ENABLED`. Set the status for `EKS_ADDON_MANAGEMENT` as `ENABLED`. GuardDuty will manage the deployment of and updates to the security agent for all the Amazon EKS clusters in your account. Alternatively, you can use the AWS CLI command by using your own regional detector ID. To find the `detectorId` for your account and current Region, see the **Settings** page in the https://console.aws.amazon.com/guardduty/ console, or run the ListDetectors API. The following example enables both `EKS_RUNTIME_MONITORING` and `EKS_ADDON_MANAGEMENT`: `aws guardduty update-detector --detector-id 12abc34d567e8fa901bc2d34e56789f0 --features '[{"Name" : "EKS_RUNTIME_MONITORING", "Status" : "ENABLED", "AdditionalConfiguration" : [{"Name" : "EKS_ADDON_MANAGEMENT", "Status" : "ENABLED"}] }]'`
Monitor all EKS clusters but exclude some of them (using exclusion tag)	1. Add a tag to the EKS cluster that you want to exclude from being monitored. The key-value pair is `GuardDutyManaged`-`false`. For more information about adding the tag, see Working with tags using the CLI, API, or eksctl in the Amazon EKS User Guide. 2. To prevent modification of tags, except by the trusted entities, use the policy provided in Prevent tags from being modified except by authorized principals in the

Preferred approach to manage GuardDuty security agent	Steps
	AWS Organizations User Guide. In this policy, replace the following details: • Replace *ec2:CreateTags* with `eks:TagResource`. • Replace *ec2:DeleteTags* with `eks:UntagResource`. • Replace *access-project* with `GuardDutyManaged` • Replace *123456789012* with the AWS account ID of the trusted entity. When you have more than one trusted entities, use the following example to add multiple `PrincipalArn`: `"aws:PrincipalArn":["arn:aws:iam::123456789012:role/org-admins/iam-admin", "arn:aws:iam::123456789012:role/org-admins/iam-admin", "arn:aws:iam::123456789012:role/org-admins/iam-admin"]` 3. **Note** Always add the exclusion tag to your EKS cluster before setting the `STATUS` of `EKS_RUNTIME_MONITORING` to `ENABLED`; otherwise, the GuardDuty security agent will be deployed on all the EKS clusters in your account. Run the updateDetector API by using your own regional detector ID and passing the `features` object

Preferred approach to manage GuardDuty security agent	Steps
	name as `EKS_RUNTIME_MONITORING` and status as `ENABLED`. Set the status for `EKS_ADDON_MANAGEMENT` as `ENABLED`. GuardDuty will manage the deployment of and updates to the security agent for all the Amazon EKS clusters that have not been excluded from being monitored. Alternatively, you can use the AWS CLI command by using your own regional detector ID. To find the `detectorId` for your account and current Region, see the **Settings** page in the https://console.aws.amazon.com/guardduty/ console, or run the ListDetectors API. The following example enables both `EKS_RUNTIME_MONITORING` and `EKS_ADDON_MANAGEMENT`: `aws guardduty update-detector --detector-id 12abc34d567e8fa901bc2d34e56789f0 --features '[{"Name" : "EKS_RUNTIME_MONITORING", "Status" : "ENABLED", "AdditionalConfiguration" : [{"Name" : "EKS_ADDON_MANAGEMENT", "Status" : "ENABLED"}] }]'`

Preferred approach to manage GuardDuty security agent	Steps
Monitor selective EKS clusters (using inclusion tag)	1. Add a tag to the EKS cluster that you want to exclude from being monitored. The key-value pair is `GuardDutyManaged -true`. For more information about adding the tag, see Working with tags using the CLI, API, or eksctl in the Amazon EKS User Guide. 2. To prevent modification of tags, except by the trusted entities, use the policy provided in Prevent tags from being modified except by authorized principals in the *AWS Organizations User Guide*. In this policy, replace the following details: • Replace *ec2:CreateTags* with `eks:TagResource` . • Replace *ec2:DeleteTags* with `eks:UntagResource` . • Replace *access-project* with `GuardDutyManaged` • Replace *123456789012* with the AWS account ID of the trusted entity. When you have more than one trusted entities, use the following example to add multiple `PrincipalArn` : `"aws:PrincipalArn":["arn:aws:iam::123456789012:role/org-admins/iam-admin", "arn:aws:iam::123456789012:role/org-admins/iam-admin", "arn:aws:iam::123456789012:role/org-admins/iam-admin"]` 3. Run the updateDetector API by using your own regional detector ID and passing the `features` object name as `EKS_RUNTIME_MONITORING` and status as `ENABLED`.

Preferred approach to manage GuardDuty security agent	Steps
	Set the status for `EKS_ADDON_MANAGEMENT` as `DISABLED`. GuardDuty will manage the deployment of and updates to the security agent for all the Amazon EKS clusters that have been tagged with the `GuardDuty Managed` -`true` pair. Alternatively, you can use the AWS CLI command by using your own regional detector ID. To find the `detectorId` for your account and current Region, see the **Settings** page in the https://console.aws.amazon.com/guardduty/ console, or run the ListDetectors API. The following example enables `EKS_RUNTIME_MONITORING` and disables `EKS_ADDON_MANAGEMENT`: `aws guardduty update-detector --detector-id 12abc34d567e8fa901bc2d34e56789f0 --features '[{"Name" : "EKS_RUNTIME_MONITORING", "Status" : "ENABLED", "AdditionalConfiguration" : [{"Name" : "EKS_ADDON_MANAGEMENT", "Status" : "DISABLED"}] }]'`

Preferred approach to manage GuardDuty security agent	Steps
Manage the security agent manually	1. Run the updateDetector API by using your own regional detector ID and passing the `features` object name as `EKS_RUNTIME_MONITORING` and status as `ENABLED`. Set the status for `EKS_ADDON_MANAGEMENT` as `DISABLED`. Alternatively, you can use the AWS CLI command by using your own regional detector ID. To find the `detectorId` for your account and current Region, see the **Settings** page in the https://console.aws.amazon.com/guardduty/ console, or run the ListDetectors API. The following example enables `EKS_RUNTIME_MONITORING` and disables `EKS_ADDON_MANAGEMENT`: `aws guardduty update-detector --detector-id 12abc34d567e8fa901bc2d34e56789f0 --features '[{"Name" : "EKS_RUNTIME_MONITORING", "Status" : "ENABLED", "AdditionalConfiguration" : [{"Name" : "EKS_ADDON_MANAGEMENT", "Status" : "ENABLED"}] }]'` 2. To manage the security agent, see Managing security agent manually for Amazon EKS cluster.

Auto-enable EKS Runtime Monitoring for all member accounts

This section includes steps to enable EKS Runtime Monitoring and manage security agent for all member accounts. This includes the delegated GuardDuty administrator account, existing member accounts, and the new accounts that join the organization.

Based on the Approaches to manage GuardDuty security agent in Amazon EKS clusters, you can choose a preferred approach and follow the steps as mentioned in the following table.

Preferred approach to manage GuardDuty security agent	Steps
Manage security agent through GuardDuty (Monitor all EKS clusters)	To selectively enable EKS Runtime Monitoring for your member accounts, run the updateMemberDetectors API operation using your own *detector ID*. Set the status for `EKS_ADDON_MANAGEMENT` as `ENABLED`. GuardDuty will manage the deployment of and updates to the security agent for all the Amazon EKS clusters in your account. Alternatively, you can use the AWS CLI command by using your own regional detector ID. To find the `detectorId` for your account and current Region, see the **Settings** page in the https://console.aws.amazon.com/guardduty/ console, or run the ListDetectors API. The following example enables both `EKS_RUNTIME_MONITORING` and `EKS_ADDON_MANAGEMENT` : `aws guardduty update-member-detectors --detector-id 12abc34d567e8fa901bc2d34e56789f0 --account-ids 111122223333 --features '[{"Name" : "EKS_RUNTIME_MONITORING", "Status" : " ENABLED", "AdditionalConfiguration" : [{"Name" : "EKS_ADDON_MANAGEMENT", "Status" : "ENABLED"}] }]'` **Note** You can also pass a list of account IDs separated by a space. When the code has successfully executed, it returns an empty list of `UnprocessedAccounts` . If there were any problems changing

Preferred approach to manage GuardDuty security agent	Steps
	the detector settings for an account, that account ID is listed along with a summary of the issue.

Preferred approach to manage GuardDuty security agent	Steps
Monitor all EKS clusters but exclude some of them (using exclusion tag)	1. Add a tag to the EKS cluster that you want to exclude from being monitored. The key-value pair is `GuardDuty Managed` -`false`. For more information about adding the tag, see [Working with tags using the CLI, API, or eksctl](#) in the Amazon EKS User Guide. 2. To prevent modification of tags, except by the trusted entities, use the policy provided in [Prevent tags from being modified except by authorized principals](#) in the *AWS Organizations User Guide*. In this policy, replace the following details: • Replace *`ec2:CreateTags`* with `eks:TagResource` . • Replace *`ec2:DeleteTags`* with `eks:UntagResource` . • Replace *`access-project`* with `GuardDutyManaged` • Replace *`123456789012`* with the AWS account ID of the trusted entity. When you have more than one trusted entities, use the following example to add multiple `PrincipalArn` : `"aws:PrincipalArn":["arn:aws:iam::123456789012:role/org-admins/iam-admin", "arn:aws:iam::123456789012:role/org-admins/iam-admin", "arn:aws:iam::123456789012:role/org-admins/iam-admin"]` 3. **Note** Always add the exclusion tag to your EKS cluster before setting the `STATUS` of `EKS_RUNTIME_MONITORING` to `ENABLED`; otherwise, the GuardDuty security agent will be deployed on all the EKS clusters in your account.

Preferred approach to manage GuardDuty security agent	Steps
	Run the updateDetector API by using your own regional detector ID and passing the `features` object name as `EKS_RUNTIME_MONITORING` and status as `ENABLED`. Set the status for `EKS_ADDON_MANAGEMENT` as `ENABLED`. GuardDuty will manage the deployment of and updates to the security agent for all the Amazon EKS clusters that have not been excluded from being monitored. Alternatively, you can use the AWS CLI command by using your own regional detector ID. To find the `detectorId` for your account and current Region, see the **Settings** page in the https://console.aws.amazon.com/guardduty/ console, or run the ListDetectors API. The following example enables both `EKS_RUNTIME_MONITORING` and `EKS_ADDON_MANAGEMENT` : `aws guardduty update-member-detectors --detector-id 12abc34d567e8fa901bc2d34e56789f0 --account-ids 111122223333 --features '[{"Name" : "EKS_RUNTIME_MONITORING", "Status" : "ENABLED", "AdditionalConfiguration" : [{"Name" : "EKS_ADDON_MANAGEMENT", "Status" : "ENABLED"}] }]'` **Note** You can also pass a list of account IDs separated by a space. When the code has successfully executed, it returns an empty list of `UnprocessedAccounts` . If there were any problems

Preferred approach to manage GuardDuty security agent	Steps
	changing the detector settings for an account, that account ID is listed along with a summary of the issue.

Preferred approach to manage GuardDuty security agent	Steps
Monitor selective EKS clusters (using inclusion tag)	1. Add a tag to the EKS cluster that you want to exclude from being monitored. The key-value pair is `GuardDuty Managed` -`true`. For more information about adding the tag, see [Working with tags using the CLI, API, or eksctl]() in the Amazon EKS User Guide. 2. To prevent modification of tags, except by the trusted entities, use the policy provided in [Prevent tags from being modified except by authorized principals]() in the *AWS Organizations User Guide*. In this policy, replace the following details: • Replace *`ec2:CreateTags`* with `eks:TagResource` . • Replace *`ec2:DeleteTags`* with `eks:UntagResource` . • Replace *`access-project`* with `GuardDutyManaged` • Replace *`123456789012`* with the AWS account ID of the trusted entity. When you have more than one trusted entities, use the following example to add multiple `PrincipalArn` : `"aws:PrincipalArn":["arn:aws:iam::123456789012:role/org-admins/iam-admin", "arn:aws:iam::123456789012:role/org-admins/iam-admin", "arn:aws:iam::123456789012:role/org-admins/iam-admin"]` 3. Run the [updateDetector]() API by using your own regional detector ID and passing the `features` object name as `EKS_RUNTIME_MONITORING` and status as `ENABLED`. Set the status for `EKS_ADDON_MANAGEMENT` as `DISABLED`. GuardDuty will manage the deployment of and updates to the security agent for all the Amazon EKS clusters that have been tagged with the `GuardDutyManaged` -`true` pair.

Preferred approach to manage GuardDuty security agent	Steps
	Alternatively, you can use the AWS CLI command by using your own regional detector ID. To find the `detectorId` for your account and current Region, see the **Settings** page in the https://console.aws.amazon.com/guardduty/ console, or run the ListDetectors API. The following example enables `EKS_RUNTIME_MONITORING` and disables `EKS_ADDON_MANAGEMENT` : `aws guardduty update-member-detectors --detector-id 12abc34d567e8fa901bc2d34e56789f0 --account-ids 111122223333 --features '[{"Name" : "EKS_RUNTIME_MONITORING", "Status" : " ENABLED", "AdditionalConfiguration" : [{"Name" : "EKS_ADDON_MANAGEMENT", "Status" : " DISABLED"}] }]'` **Note** You can also pass a list of account IDs separated by a space. When the code has successfully executed, it returns an empty list of `UnprocessedAccounts` . If there were any problems changing the detector settings for an account, that account ID is listed along with a summary of the issue.

Preferred approach to manage GuardDuty security agent	Steps
Manage the security agent manually	1. Run the updateDetector API by using your own regional detector ID and passing the `features` object name as `EKS_RUNTIME_MONITORING` and status as `ENABLED`. Set the status for `EKS_ADDON_MANAGEMENT` as `DISABLED`. Alternatively, you can use the AWS CLI command by using your own regional detector ID. To find the `detectorId` for your account and current Region, see the **Settings** page in the https://console.aws.amazon.com/guardduty/ console, or run the ListDetectors API. The following example enables `EKS_RUNTIME_MONITORING` and disables `EKS_ADDON_MANAGEMENT`: `aws guardduty update-member-detectors --detector-id 12abc34d567e8fa901bc2d34e56789f0 --account-ids 555555555555 --features '[{"Name" : "EKS_RUNTIME_MONITORING", "Status" : " ENABLED", "AdditionalConfiguration" : [{"Name" : "EKS_ADDON_MANAGEMENT", "Status" : " ENABLED"}] } ]'` 2. To manage the security agent, see Managing security agent manually for Amazon EKS cluster.

Configuring EKS Runtime Monitoring for all existing active member accounts

This section includes the steps to enable EKS Runtime Monitoring and manage GuardDuty security agent for existing active member accounts in your organization.

Based on the Approaches to manage GuardDuty security agent in Amazon EKS clusters, you can choose a preferred approach and follow the steps as mentioned in the following table.

<table>
<tr><th>Preferred approach to manage GuardDuty security agent</th><th>Steps</th></tr>
<tr><td>Manage security agent through GuardDuty (Monitor all EKS clusters)</td><td>

To selectively enable EKS Runtime Monitoring for your member accounts, run the updateMemberDetectors API operation using your own *detector ID*.

Set the status for `EKS_ADDON_MANAGEMENT` as `ENABLED`.

GuardDuty will manage the deployment of and updates to the security agent for all the Amazon EKS clusters in your account.

Alternatively, you can use the AWS CLI command by using your own regional detector ID. To find the `detectorId` for your account and current Region, see the **Settings** page in the https://console.aws.amazon.com/guardduty/ console, or run the ListDetectors API.

The following example enables both `EKS_RUNTIME_MONITORING` and `EKS_ADDON_MANAGEMENT`:

```
aws guardduty update-member-detectors --detector-id 12abc34d567e8fa901bc2d34e56789f0 --account-ids 111122223333 --features '[{"Name" : "EKS_RUNTIME_MONITORING", "Status" : "ENABLED", "AdditionalConfiguration" : [{"Name" : "EKS_ADDON_MANAGEMENT", "Status" : "ENABLED"}] }]'
```

Note

You can also pass a list of account IDs separated by a space.

When the code has successfully executed, it returns an empty list of `UnprocessedAccounts`. If there were any problems changing the detector settings for an account, that account ID is listed along with a summary of the issue.

</td></tr>
</table>

<table>
<tr><th>Preferred approach to manage GuardDuty security agent</th><th>Steps</th></tr>
<tr><td>Monitor all EKS clusters but exclude some of them (using exclusion tag)</td><td>

Add a tag to the EKS cluster that you want to exclude from being monitored. The key-value pair is <code>GuardDuty Managed</code> -<code>false</code>. For more information about adding the tag, see Working with tags using the CLI, API, or eksctl in the Amazon EKS User Guide.
To prevent modification of tags, except by the trusted entities, use the policy provided in Prevent tags from being modified except by authorized principals in the AWS Organizations User Guide. In this policy, replace the following details:

Replace <code>ec2:CreateTags</code> with <code>eks:TagResource</code> .
Replace <code>ec2:DeleteTags</code> with <code>eks:UntagResource</code> .
Replace <code>access-project</code> with <code>GuardDutyManaged</code>
Replace <code>123456789012</code> with the AWS account ID of the trusted entity.

When you have more than one trusted entities, use the following example to add multiple <code>PrincipalArn</code> :
<pre>"aws:PrincipalArn":["arn:aws:iam::123456789012:role/org-admins/iam-admin", "arn:aws:iam::123456789012:role/org-admins/iam-admin", "arn:aws:iam::123456789012:role/org-admins/iam-admin"]</pre>

Note
Always add the exclusion tag to your EKS cluster before setting the <code>STATUS</code> of <code>EKS_RUNTIME_MONITORING</code> to <code>ENABLED</code>; otherwise, the GuardDuty security agent will be deployed on all the EKS clusters in your account.

</td></tr>
</table>

Preferred approach to manage GuardDuty security agent	Steps
	To selectively enable EKS Runtime Monitoring for your member accounts, run the [updateMemberDetectors](#) API operation using your own *detector ID*. Set the status for `EKS_ADDON_MANAGEMENT` as `ENABLED`. GuardDuty will manage the deployment of and updates to the security agent for all the Amazon EKS clusters that have not been excluded from being monitored. Alternatively, you can use the AWS CLI command by using your own regional detector ID. To find the `detectorId` for your account and current Region, see the **Settings** page in the https://console.aws.amazon.com/guardduty/ console, or run the [ListDetectors](#) API. The following example enables both `EKS_RUNTIME_MONITORING` and `EKS_ADDON_MANAGEMENT`: `aws guardduty update-member-detectors --detector-id 12abc34d567e8fa901bc2d34e56789f0 --account-ids 111122223333 --features '[{"Name" : "EKS_RUNTIME_MONITORING", "Status" : "ENABLED", "AdditionalConfiguration" : [{"Name" : "EKS_ADDON_MANAGEMENT", "Status" : "ENABLED"}] }]'` **Note** You can also pass a list of account IDs separated by a space. When the code has successfully executed, it returns an empty list of `UnprocessedAccounts`. If there were any problems

Preferred approach to manage GuardDuty security agent	Steps
	changing the detector settings for an account, that account ID is listed along with a summary of the issue.

Preferred approach to manage GuardDuty security agent	Steps
Monitor selective EKS clusters (using inclusion tag)	1. Add a tag to the EKS cluster that you want to exclude from being monitored. The key-value pair is `GuardDuty Managed` -`true`. For more information about adding the tag, see Working with tags using the CLI, API, or eksctl in the Amazon EKS User Guide. 2. To prevent modification of tags, except by the trusted entities, use the policy provided in Prevent tags from being modified except by authorized principals in the *AWS Organizations User Guide*. In this policy, replace the following details: • Replace *`ec2:CreateTags`* with `eks:TagResource` . • Replace *`ec2:DeleteTags`* with `eks:UntagResource` . • Replace *`access-project`* with `GuardDutyManaged` • Replace *`123456789012`* with the AWS account ID of the trusted entity. When you have more than one trusted entities, use the following example to add multiple `PrincipalArn` : `"aws:PrincipalArn":["arn:aws:iam::123456789012:role/org-admins/iam-admin", "arn:aws:iam::123456789012:role/org-admins/iam-admin", "arn:aws:iam::123456789012:role/org-admins/iam-admin"]` 3. To selectively enable EKS Runtime Monitoring for your member accounts, run the updateMemberDetectors API operation using your own *`detector ID`*. Set the status for `EKS_ADDON_MANAGEMENT` as `DISABLED`. GuardDuty will manage the deployment of and updates to the security agent for all the Amazon EKS clusters that have been tagged with the `GuardDutyManaged` -`true` pair.

Preferred approach to manage GuardDuty security agent	Steps
	Alternatively, you can use the AWS CLI command by using your own regional detector ID. To find the `detectorId` for your account and current Region, see the **Settings** page in the https://console.aws.amazon.com/guardduty/ console, or run the ListDetectors API. The following example enables `EKS_RUNTIME_MONITORING` and disables `EKS_ADDON_MANAGEMENT` : `aws guardduty update-member-detectors --detector-id 12abc34d567e8fa901bc2d34e56789f0 --account-ids 111122223333 --features '[{"Name" : "EKS_RUNTIME_MONITORING", "Status" : " ENABLED", "AdditionalConfiguration" : [{"Name" : "EKS_ADDON_MANAGEMENT", "Status" : " DISABLED"}] }]'` **Note** You can also pass a list of account IDs separated by a space. When the code has successfully executed, it returns an empty list of `UnprocessedAccounts` . If there were any problems changing the detector settings for an account, that account ID is listed along with a summary of the issue.

Preferred approach to manage GuardDuty security agent	Steps
Manage the security agent manually	1. To selectively enable EKS Runtime Monitoring for your member accounts, run the updateMemberDetectors API operation using your own *detector ID*. Set the status for `EKS_ADDON_MANAGEMENT` as `DISABLED`. Alternatively, you can use the AWS CLI command by using your own regional detector ID. To find the `detectorId` for your account and current Region, see the **Settings** page in the https://console.aws.amazon.com/guardduty/ console, or run the ListDetectors API. The following example enables `EKS_RUNTIME_MONITORING` and disables `EKS_ADDON_MANAGEMENT` : `aws guardduty update-member-detectors --detector-id 12abc34d567e8fa901bc2d34e56789f0 --account-ids 555555555555 --features '[{"Name" : "EKS_RUNTIME_MONITORING", "Status" : " ENABLED", "AdditionalConfiguration" : [{"Name" : "EKS_ADDON_MANAGEMENT", "Status" : " ENABLED"}] } ]'` 2. To manage the security agent, see Managing security agent manually for Amazon EKS cluster.

Auto-enable EKS Runtime Monitoring for new members

The delegated GuardDuty administrator account can auto-enable EKS Runtime Monitoring and choose an approach for how to manage the GuardDuty security agent for new accounts that join your organization.

Based on the Approaches to manage GuardDuty security agent in Amazon EKS clusters, you can choose a preferred approach and follow the steps as mentioned in the following table.

Preferred approach to manage GuardDuty security agent	Steps
Manage security agent through GuardDuty (Monitor all EKS clusters)	To selectively enable EKS Runtime Monitoring for your new accounts, invoke the UpdateOrganizationConfiguration API operation using your own *detector ID*. Set the status for `EKS_ADDON_MANAGEMENT` as `ENABLED`. GuardDuty will manage the deployment of and updates to the security agent for all the Amazon EKS clusters in your account. Alternatively, you can use the AWS CLI command by using your own regional detector ID. To find the `detectorId` for your account and current Region, see the **Settings** page in the https://console.aws.amazon.com/guardduty/ console, or run the ListDetectors API. The following example enables both `EKS_RUNTIME_MONITORING` and `EKS_ADDON_MANAGEMENT` for a single account. You can also pass a list of account IDs separated by a space. To find the `detectorId` for your account and current Region, see the **Settings** page in the https://console.aws.amazon.com/guardduty/ console, or run the ListDetectors API. `aws guardduty update-organization-configuration --detector-id 12abc34d567e8fa901bc2d34e56789f0 --autoEnable --features '[{"Name" : "EKS_RUNTIME_MONITORING", "AutoEnable": "NEW", "AdditionalConfiguration" : [{"Name" : "EKS_ADDON_MANAGEMENT", "AutoEnable": "NEW"}] }]'` When the code has successfully executed, it returns an empty list of `UnprocessedAccounts`. If there were any

Preferred approach to manage GuardDuty security agent	Steps
	problems changing the detector settings for an account, that account ID is listed along with a summary of the issue.

Preferred approach to manage GuardDuty security agent	Steps
Monitor all EKS clusters but exclude some of them (using exclusion tag)	1. Add a tag to the EKS cluster that you want to exclude from being monitored. The key-value pair is `GuardDutyManaged -false`. For more information about adding the tag, see Working with tags using the CLI, API, or eksctl in the Amazon EKS User Guide. 2. To prevent modification of tags, except by the trusted entities, use the policy provided in Prevent tags from being modified except by authorized principals in the *AWS Organizations User Guide*. In this policy, replace the following details: • Replace *`ec2:CreateTags`* with `eks:TagResource` . • Replace *`ec2:DeleteTags`* with `eks:UntagResource` . • Replace *`access-project`* with `GuardDutyManaged` • Replace *`123456789012`* with the AWS account ID of the trusted entity. When you have more than one trusted entities, use the following example to add multiple `PrincipalArn` : `"aws:PrincipalArn":["arn:aws:iam::123456789012:role/org-admins/iam-admin", "arn:aws:iam::123456789012:role/org-admins/iam-admin", "arn:aws:iam::123456789012:role/org-admins/iam-admin"]` 3. **Note** Always add the exclusion tag to your EKS cluster before setting the STATUS of

Preferred approach to manage GuardDuty security agent	Steps
	EKS_RUNTIME_MONITORING to ENABLED; otherwise, the GuardDuty security agent will be deployed on all the EKS clusters in your account. To selectively enable EKS Runtime Monitoring for your new accounts, invoke the [UpdateOrganizationConfiguration](#) API operation using your own *detector ID*. Set the status for EKS_ADDON_MANAGEMENT as ENABLED. GuardDuty will manage the deployment of and updates to the security agent for all the Amazon EKS clusters that have not been excluded from being monitored. Alternatively, you can use the AWS CLI command by using your own regional detector ID. To find the detectorId for your account and current Region, see the **Settings** page in the https://console.aws.amazon.com/guardduty/ console, or run the [ListDetectors](#) API. The following example enables both EKS_RUNTIME_MONITORING and EKS_ADDON_MANAGEMENT for a single account. You can also pass a list of account IDs separated by a space. To find the detectorId for your account and current Region, see the **Settings** page in the https://console.aws.amazon.com/guardduty/ console, or run the [ListDetectors](#) API.

Preferred approach to manage GuardDuty security agent	Steps
	`aws guardduty update-organization-configuration --detector-id 12abc34d567e8fa901bc2d34e56789f0 --autoEnable --features '[{"Name" : "EKS_RUNTIME_MONITORING", "AutoEnable": "NEW", "AdditionalConfiguration" : [{"Name" : "EKS_ADDON_MANAGEMENT", "AutoEnable": "NEW"}] }]'` When the code has successfully executed, it returns an empty list of `UnprocessedAccounts`. If there were any problems changing the detector settings for an account, that account ID is listed along with a summary of the issue.

Preferred approach to manage GuardDuty security agent	Steps
Monitor selective EKS clusters (using inclusion tag)	1. Add a tag to the EKS cluster that you want to exclude from being monitored. The key-value pair is `GuardDutyManaged` -`true`. For more information about adding the tag, see [Working with tags using the CLI, API, or eksctl](#) in the Amazon EKS User Guide. 2. To prevent modification of tags, except by the trusted entities, use the policy provided in [Prevent tags from being modified except by authorized principals](#) in the *AWS Organizations User Guide*. In this policy, replace the following details: • Replace *`ec2:CreateTags`* with `eks:TagResource` . • Replace *`ec2:DeleteTags`* with `eks:UntagResource` . • Replace *`access-project`* with `GuardDutyManaged` • Replace *`123456789012`* with the AWS account ID of the trusted entity. When you have more than one trusted entities, use the following example to add multiple `PrincipalArn` : `"aws:PrincipalArn":["arn:aws:iam::123456789012:role/org-admins/iam-admin", "arn:aws:iam::123456789012:role/org-admins/iam-admin", "arn:aws:iam::123456789012:role/org-admins/iam-admin"]` 3. To selectively enable EKS Runtime Monitoring for your new accounts, invoke the [UpdateOrganizationConfiguration](#) API operation using your own *`detector ID`*.

Preferred approach to manage GuardDuty security agent	Steps
	Set the status for `EKS_ADDON_MANAGEMENT` as `DISABLED`. GuardDuty will manage the deployment of and updates to the security agent for all the Amazon EKS clusters that have been tagged with the `GuardDuty Managed` -`true` pair. Alternatively, you can use the AWS CLI command by using your own regional detector ID. To find the `detectorId` for your account and current Region, see the **Settings** page in the https://console.aws.amazon.com/guardduty/ console, or run the ListDetectors API. The following example enables `EKS_RUNTIME_MONITORING` and disables `EKS_ADDON_MANAGEMENT` for a single account. You can also pass a list of account IDs separated by a space. To find the `detectorId` for your account and current Region, see the **Settings** page in the https://console.aws.amazon.com/guardduty/ console, or run the ListDetectors API. `aws guardduty update-organization-configuration --detector-id 12abc34d567e8fa901bc2d34e56789f0 --autoEnable --features '[{"Name" : "EKS_RUNTIME_MONITORING", "AutoEnable": "NEW", "AdditionalConfiguration" : [{"Name" : "EKS_ADDON_MANAGEMENT", "AutoEnable": "NEW"}] }]'` When the code has successfully executed, it returns an empty list of `UnprocessedAccounts`. If there

Preferred approach to manage GuardDuty security agent	Steps
	were any problems changing the detector settings for an account, that account ID is listed along with a summary of the issue.

Preferred approach to manage GuardDuty security agent	Steps
Manage the security agent manually	1. To selectively enable EKS Runtime Monitoring for your new accounts, invoke the [UpdateOrganizationConfiguration](#) API operation using your own *detector ID*. Set the status for `EKS_ADDON_MANAGEMENT` as `DISABLED`. Alternatively, you can use the AWS CLI command by using your own regional detector ID. To find the `detectorId` for your account and current Region, see the **Settings** page in the https://console.aws.amazon.com/guardduty/ console, or run the [ListDetectors](#) API. The following example enables `EKS_RUNTIME_MONITORING` and disables `EKS_ADDON_MANAGEMENT` for a single account. You can also pass a list of account IDs separated by a space. To find the `detectorId` for your account and current Region, see the **Settings** page in the https://console.aws.amazon.com/guardduty/ console, or run the [ListDetectors](#) API. `aws guardduty update-organization-configuration --detector-id 12abc34d567e8fa901bc2d34e56789f0 --autoEnable --features '[{"Name" : "EKS_RUNTIME_MONITORING", "AutoEnable": "NEW", "AdditionalConfiguration" : [{"Name" : "EKS_ADDON_MANAGEMENT", "AutoEnable": "NEW"}] }]'` When the code has successfully executed, it returns an empty list of `UnprocessedAccounts`. If there were any problems changing the detector settings

Preferred approach to manage GuardDuty security agent	Steps
	for an account, that account ID is listed along with a summary of the issue. 2. To manage the security agent, see Managing security agent manually for Amazon EKS cluster.

Enable EKS Runtime Monitoring for individual active member accounts

This section includes the steps to configure EKS Runtime Monitoring and manage security agent for individual active member accounts.

Based on the Approaches to manage GuardDuty security agent in Amazon EKS clusters, you can choose a preferred approach and follow the steps as mentioned in the following table.

Preferred approach to manage GuardDuty security agent	Steps
Manage security agent through GuardDuty (Monitor all EKS clusters)	To selectively enable EKS Runtime Monitoring for your member accounts, run the updateMemberDetectors API operation using your own *`detector ID`*. Set the status for `EKS_ADDON_MANAGEMENT` as `ENABLED`. GuardDuty will manage the deployment of and updates to the security agent for all the Amazon EKS clusters in your account. Alternatively, you can use the AWS CLI command by using your own regional detector ID. To find the `detectorId` for your account and current Region, see the **Settings** page in the https://console.aws.amazon.com/guardduty/ console, or run the ListDetectors API. The following example enables both `EKS_RUNTIME_MONITORING` and `EKS_ADDON_MANAGEMENT`:

Preferred approach to manage GuardDuty security agent	Steps
	`aws guardduty update-member-detectors --detector-id 12abc34d567e8fa901bc2d34e56789f0 --account-ids 111122223333 --features '[{"Name" : "EKS_RUNTIME_MONITORING", "Status" : " ENABLED", "AdditionalConfiguration" : [{"Name" : "EKS_ADDON_MANAGEMENT", "Status" : " ENABLED"}] }]'` **Note** You can also pass a list of account IDs separated by a space. When the code has successfully executed, it returns an empty list of `UnprocessedAccounts` . If there were any problems changing the detector settings for an account, that account ID is listed along with a summary of the issue.

<table>
<tr><th>Preferred approach to manage GuardDuty security agent</th><th>Steps</th></tr>
<tr><td>Monitor all EKS clusters but exclude some of them (using exclusion tag)</td><td>

Add a tag to the EKS cluster that you want to exclude from being monitored. The key-value pair is <code>GuardDutyManaged -false</code>. For more information about adding the tag, see Working with tags using the CLI, API, or eksctl in the Amazon EKS User Guide.
To prevent modification of tags, except by the trusted entities, use the policy provided in Prevent tags from being modified except by authorized principals in the AWS Organizations User Guide. In this policy, replace the following details:

Replace ec2:CreateTags with <code>eks:TagResource</code> .
Replace ec2:DeleteTags with <code>eks:UntagResource</code> .
Replace access-project with <code>GuardDutyManaged</code>
Replace 123456789012 with the AWS account ID of the trusted entity.
When you have more than one trusted entities, use the following example to add multiple <code>PrincipalArn</code> :
<pre>"aws:PrincipalArn":["arn:aws:iam::123456789012:role/org-admins/iam-admin", "arn:aws:iam::123456789012:role/org-admins/iam-admin", "arn:aws:iam::123456789012:role/org-admins/iam-admin"]</pre>

Note
Always add the exclusion tag to your EKS cluster before setting the STATUS of

</td></tr>
</table>

Preferred approach to manage GuardDuty security agent	Steps
	EKS_RUNTIME_MONITORING to ENABLED; otherwise, the GuardDuty security agent will be deployed on all the EKS clusters in your account. To selectively enable EKS Runtime Monitoring for your member accounts, run the updateMemberDetectors API operation using your own *detector ID*. Set the status for EKS_ADDON_MANAGEMENT as ENABLED. GuardDuty will manage the deployment of and updates to the security agent for all the Amazon EKS clusters that have not been excluded from being monitored. Alternatively, you can use the AWS CLI command by using your own regional detector ID. To find the detectorId for your account and current Region, see the **Settings** page in the https://console.aws.amazon.com/guardduty/ console, or run the ListDetectors API. The following example enables both EKS_RUNTIME_MONITORING and EKS_ADDON_MANAGEMENT : `aws guardduty update-member-detectors --detector-id 12abc34d567e8fa901bc2d34e56789f0 --account-ids 111122223333 --features '[{"Name" : "EKS_RUNTIME_MONITORING", "Status" : "ENABLED", "AdditionalConfiguration" : [{"Name" : "EKS_ADDON_MANAGEMENT", "Status" : "ENABLED"}] }]'`

Preferred approach to manage GuardDuty security agent	Steps
	Note You can also pass a list of account IDs separated by a space. When the code has successfully executed, it returns an empty list of `UnprocessedAccounts` . If there were any problems changing the detector settings for an account, that account ID is listed along with a summary of the issue.

Preferred approach to manage GuardDuty security agent	Steps
Monitor selective EKS clusters (using inclusion tag)	1. Add a tag to the EKS cluster that you want to exclude from being monitored. The key-value pair is `GuardDutyManaged` -`true`. For more information about adding the tag, see Working with tags using the CLI, API, or eksctl in the Amazon EKS User Guide. 2. To prevent modification of tags, except by the trusted entities, use the policy provided in Prevent tags from being modified except by authorized principals in the *AWS Organizations User Guide*. In this policy, replace the following details: • Replace *`ec2:CreateTags`* with `eks:TagResource` . • Replace *`ec2:DeleteTags`* with `eks:UntagResource` . • Replace *`access-project`* with `GuardDutyManaged` • Replace *`123456789012`* with the AWS account ID of the trusted entity. When you have more than one trusted entities, use the following example to add multiple `PrincipalArn` : `"aws:PrincipalArn":["arn:aws:iam::123456789012:role/org-admins/iam-admin", "arn:aws:iam::123456789012:role/org-admins/iam-admin", "arn:aws:iam::123456789012:role/org-admins/iam-admin"]` 3. To selectively enable EKS Runtime Monitoring for your member accounts, run the updateMemberDetectors API operation using your own *`detector ID`*.

Preferred approach to manage GuardDuty security agent	Steps
	Set the status for EKS_ADDON_MANAGEMENT as DISABLED. GuardDuty will manage the deployment of and updates to the security agent for all the Amazon EKS clusters that have been tagged with the GuardDuty Managed -true pair. Alternatively, you can use the AWS CLI command by using your own regional detector ID. To find the detectorId for your account and current Region, see the **Settings** page in the https://console.aws.amazon.com/guardduty/ console, or run the ListDetectors API. The following example enables EKS_RUNTIME_MONITORING and disables EKS_ADDON_MANAGEMENT : `aws guardduty update-member-detectors --detector-id 12abc34d567e8fa901bc2d34e56789f0 --account-ids 111122223333 --features '[{"Name" : "EKS_RUNTIME_MONITORING", "Status" : "ENABLED", "AdditionalConfiguration" : [{"Name" : "EKS_ADDON_MANAGEMENT", "Status" : "DISABLED"}] }]'` **Note** You can also pass a list of account IDs separated by a space. When the code has successfully executed, it returns an empty list of UnprocessedAccounts . If there

Preferred approach to manage GuardDuty security agent	Steps
	were any problems changing the detector settings for an account, that account ID is listed along with a summary of the issue.
Manage the security agent manually	1. To selectively enable EKS Runtime Monitoring for your member accounts, run the [updateMemberDetectors](#) API operation using your own *detector ID*. Set the status for `EKS_ADDON_MANAGEMENT` as `DISABLED`. Alternatively, you can use the AWS CLI command by using your own regional detector ID. To find the `detectorId` for your account and current Region, see the **Settings** page in the https://console.aws.amazon.com/guardduty/ console, or run the [ListDetectors](#) API. The following example enables `EKS_RUNTIME_MONITORING` and disables `EKS_ADDON_MANAGEMENT`: `aws guardduty update-member-detectors --detector-id 12abc34d567e8fa901bc2d34e56789f0 --account-ids 555555555555 --features '[{"Name" : "EKS_RUNTIME_MONITORING", "Status" : "ENABLED", "AdditionalConfiguration" : [{"Name" : "EKS_ADDON_MANAGEMENT", "Status" : "ENABLED"}] } ]'` 2. To manage the security agent, see [Managing security agent manually for Amazon EKS cluster](#).

Configuring EKS Runtime Monitoring for a standalone account (API)

A standalone account owns the decision to enable or disable a protection plan in their AWS account in a specific AWS Region.

If your account is associated with a GuardDuty administrator account through AWS Organizations, or by the method of invitation, this section doesn't apply to your account. For more information, see Configuring EKS Runtime Monitoring for multiple-account environments (API).

After you enable Runtime Monitoring, ensure to install GuardDuty security agent through automated configuration or manual deployment. As a part of completing all the steps listed in the following procedure, make sure to install the security agent.

Based on the Approaches to manage GuardDuty security agent in Amazon EKS clusters, you can choose a preferred approach and follow the steps as mentioned in the following table.

Preferred approach to manage GuardDuty security agent	Steps
Manage security agent through GuardDuty (Monitor all EKS clusters)	1. Run the updateDetector API by using your own regional detector ID and passing the `features` object name as `EKS_RUNTIME_MONITORING` and status as `ENABLED`. Set the status for `EKS_ADDON_MANAGEMENT` as `ENABLED`. GuardDuty will manage the deployment of and updates to the security agent for all the Amazon EKS clusters in your account. 2. Alternatively, you can use the AWS CLI command by using your own regional detector ID. To find the `detectorId` for your account and current Region, see the **Settings** page in the https://console.aws.amazon.com/guardduty/ console, or run the ListDetectors API. The following example enables both `EKS_RUNTIME_MONITORING` and `EKS_ADDON_MANAGEMENT`:

Preferred approach to manage GuardDuty security agent	Steps
	`aws guardduty update-detector --detector-id 12abc34d567e8fa901bc2d34e56789f0 --features '[{"Name" : "EKS_RUNTIME_MONITORING", "Status" : " ENABLED", "AdditionalConfiguration" : [{"Name" : "EKS_ADDON_MANAGEMENT", "Status" : " ENABLED"}] }]'`

<table>
<tr><th>Preferred approach to manage GuardDuty security agent</th><th>Steps</th></tr>
<tr><td>Monitor all EKS clusters but exclude some of them (using exclusion tag)</td><td>

Add a tag to the EKS cluster that you want to exclude from being monitored. The key-value pair is <code>GuardDutyManaged -false</code>. For more information about adding the tag, see Working with tags using the CLI, API, or eksctl in the Amazon EKS User Guide.
To prevent modification of tags, except by the trusted entities, use the policy provided in Prevent tags from being modified except by authorized principals in the AWS Organizations User Guide. In this policy, replace the following details:

Replace ec2:CreateTags with <code>eks:TagResource</code> .
Replace ec2:DeleteTags with <code>eks:UntagResource</code> .
Replace access-project with <code>GuardDutyManaged</code>
Replace 123456789012 with the AWS account ID of the trusted entity.
When you have more than one trusted entities, use the following example to add multiple <code>PrincipalArn</code> :
<pre>"aws:PrincipalArn":["arn:aws:iam::123456789012:role/org-admins/iam-admin", "arn:aws:iam::123456789012:role/org-admins/iam-admin", "arn:aws:iam::123456789012:role/org-admins/iam-admin"]</pre>

Note
Always add the exclusion tag to your EKS cluster before setting the STATUS of

</td></tr>
</table>

Preferred approach to manage GuardDuty security agent	Steps
	EKS_RUNTIME_MONITORING to ENABLED; otherwise, the GuardDuty security agent will be deployed on all the EKS clusters in your account. Run the updateDetector API by using your own regional detector ID and passing the `features` object name as `EKS_RUNTIME_MONITORING` and status as `ENABLED`. Set the status for `EKS_ADDON_MANAGEMENT` as `ENABLED`. GuardDuty will manage the deployment of and updates to the security agent for all the Amazon EKS clusters that have not been excluded from being monitored. Alternatively, you can use the AWS CLI command by using your own regional detector ID. To find the `detectorId` for your account and current Region, see the **Settings** page in the https://console.aws.amazon.com/guardduty/ console, or run the ListDetectors API. The following example enables both `EKS_RUNTIME_MONITORING` and `EKS_ADDON_MANAGEMENT`: `aws guardduty update-detector --detector-id 12abc34d567e8fa901bc2d34e56789f0 --features '[{"Name" : "EKS_RUNTIME_MONITORING", "Status" : "ENABLED", "AdditionalConfiguration" : [{"Name" : "EKS_ADDON_MANAGEMENT", "Status" : "ENABLED"}] }]'`

<table>
<tr><th>Preferred approach to manage GuardDuty security agent</th><th>Steps</th></tr>
<tr><td>Monitor selective EKS clusters (using inclusion tag)</td><td>

Add a tag to the EKS cluster that you want to exclude from being monitored. The key-value pair is <code>GuardDutyManaged</code> -<code>true</code>. For more information about adding the tag, see Working with tags using the CLI, API, or eksctl in the Amazon EKS User Guide.
To prevent modification of tags, except by the trusted entities, use the policy provided in Prevent tags from being modified except by authorized principals in the AWS Organizations User Guide. In this policy, replace the following details:

Replace <code>ec2:CreateTags</code> with <code>eks:TagResource</code> .
Replace <code>ec2:DeleteTags</code> with <code>eks:UntagResource</code> .
Replace <code>access-project</code> with <code>GuardDutyManaged</code>
Replace <code>123456789012</code> with the AWS account ID of the trusted entity.

When you have more than one trusted entities, use the following example to add multiple <code>PrincipalArn</code> :

<pre><code>"aws:PrincipalArn":["arn:aws:iam::123456789012:role/org-admins/iam-admin", "arn:aws:iam::123456789012:role/org-admins/iam-admin", "arn:aws:iam::123456789012:role/org-admins/iam-admin"]</code></pre>

Run the updateDetector API by using your own regional detector ID and passing the <code>features</code> object name as <code>EKS_RUNTIME_MONITORING</code> and status as <code>ENABLED</code>.

</td></tr>
</table>

Preferred approach to manage GuardDuty security agent	Steps
	Set the status for EKS_ADDON_MANAGEMENT as DISABLED. GuardDuty will manage the deployment of and updates to the security agent for all the Amazon EKS clusters that have been tagged with the GuardDuty Managed -true pair. Alternatively, you can use the AWS CLI command by using your own regional detector ID. To find the detectorId for your account and current Region, see the **Settings** page in the https://console.aws.amazon.com/guardduty/ console, or run the ListDetectors API. The following example enables EKS_RUNTIME_MONITORING and disables EKS_ADDON_MANAGEMENT : `aws guardduty update-detector --detector-id 12abc34d567e8fa901bc2d34e56789f0 --features '[{"Name" : "EKS_RUNTIME_MONITORING", "Status" : " ENABLED", "AdditionalConfiguration" : [{"Name" : "EKS_ADDON_MANAGEMENT", "Status" : " DISABLED"}] }]'`

Preferred approach to manage GuardDuty security agent	Steps
Manage the security agent manually	1. Run the updateDetector API by using your own regional detector ID and passing the `features` object name as `EKS_RUNTIME_MONITORING` and status as `ENABLED`. Set the status for `EKS_ADDON_MANAGEMENT` as `DISABLED`. Alternatively, you can use the AWS CLI command by using your own regional detector ID. To find the `detectorId` for your account and current Region, see the **Settings** page in the https://console.aws.amazon.com/guardduty/ console, or run the ListDetectors API. The following example enables `EKS_RUNTIME_MONITORING` and disables `EKS_ADDON_MANAGEMENT`: `aws guardduty update-detector --detector-id 12abc34d567e8fa901bc2d34e56789f0 --features '[{"Name" : "EKS_RUNTIME_MONITORING", "Status" : " ENABLED", "AdditionalConfiguration" : [{"Name" : "EKS_ADDON_MANAGEMENT", "Status" : " DISABLED"}] }]'` 2. To manage the security agent, see Managing security agent manually for Amazon EKS cluster.

Migrating from EKS Runtime Monitoring to Runtime Monitoring

With the launch of GuardDuty Runtime Monitoring, the threat detection coverage has been expanded to Amazon ECS containers and Amazon EC2 instances. EKS Runtime Monitoring experience has now been consolidated into Runtime Monitoring. You can enable Runtime Monitoring and manage individual GuardDuty security agents for each resource type (Amazon EC2 instance, Amazon ECS cluster, and Amazon EKS cluster) for which you want to monitor the runtime behavior.

GuardDuty has consolidated the console experience for EKS Runtime Monitoring into Runtime Monitoring. GuardDuty recommends Checking EKS Runtime Monitoring configuration status and Migrating from EKS Runtime Monitoring to Runtime Monitoring.

As a part of migrating to Runtime Monitoring, ensure to Disable EKS Runtime Monitoring. This is important because if you later choose to disable Runtime Monitoring and you do not disable EKS Runtime Monitoring, you will continue incurring usage cost for EKS Runtime Monitoring.

To migrate from EKS Runtime Monitoring to Runtime Monitoring

1. The GuardDuty console supports EKS Runtime Monitoring as a part of Runtime Monitoring.

   You can start using Runtime Monitoring by Checking EKS Runtime Monitoring configuration status of your organization and accounts.

   Make sure to not disable EKS Runtime Monitoring before enabling Runtime Monitoring. If you disable EKS Runtime Monitoring, the Amazon EKS add-on management will also get disabled. Continue with the following steps in the listed order.
2. Make sure you meet all the Prerequisites to enabling Runtime Monitoring.
3. Enable Runtime Monitoring by replicating the same organization configuration settings for Runtime Monitoring as you have for EKS Runtime Monitoring. For more information, see Enabling Runtime Monitoring.
   - If you have a standalone account, you need to enable Runtime Monitoring.

     If your GuardDuty security agent is deployed already, the corresponding settings are replicated automatically and you don't need to configure the settings again.
   - If you have an organization with auto-enablement settings, make sure to replicate the same auto-enablement settings for Runtime Monitoring.
   - If you have an organization with settings configured for existing active member accounts individually, make sure to enable Runtime Monitoring and configure the GuardDuty security agent for these members individually.
4. After you have ensured that the Runtime Monitoring and GuardDuty security agent settings are correct, disable EKS Runtime Monitoring by using either the API or the AWS CLI command.
5. (Optional) if you want to clean any resource associated with the GuardDuty security agent, see Disabling, uninstalling, and cleaning up resources in Runtime Monitoring.

If you want to continue using EKS Runtime Monitoring without enabling Runtime Monitoring, see EKS Runtime Monitoring in GuardDuty. Based on your use case, choose the steps to configure EKS Runtime Monitoring for a standalone account or for multiple member accounts.

Checking EKS Runtime Monitoring configuration status

Use the following APIs or AWS CLI commands to check the existing configuration status of EKS Runtime Monitoring.

To check existing EKS Runtime Monitoring configuration status in your account

- Run GetDetector to check the configuration status of your own account.
- Alternatively, you can run the following command by using AWS CLI:

```
aws guardduty get-detector --detector-id 12abc34d567e8fa901bc2d34e56789f0 --
region us-east-1
```

  Make sure to replace the detector ID of your AWS account and the current Region. To find the `detectorId` for your account and current Region, see the **Settings** page in the https://console.aws.amazon.com/guardduty/ console, or run the ListDetectors API.

To check existing EKS Runtime Monitoring configuration status for your organization (as a delegated GuardDuty administrator account only)

- Run DescribeOrganizationConfiguration to check the configuration status of your organization.

  Alternatively, you can run the following command using AWS CLI:

```
aws guardduty describe-organization-configuration --detector-
id 12abc34d567e8fa901bc2d34e56789f0 --region us-east-1
```

  Make sure to replace the detector ID with the detector ID of your delegated GuardDuty administrator account and the Region with your current Region. To find the `detectorId` for your account and current Region, see the **Settings** page in the https://console.aws.amazon.com/guardduty/ console, or run the ListDetectors API.

Disabling EKS Runtime Monitoring after migrating to Runtime Monitoring

After you have ensured that the existing settings for your account or organization have been replicated to Runtime Monitoring, you can disable EKS Runtime Monitoring.

To disable EKS Runtime Monitoring

- **To disable EKS Runtime Monitoring in your own account**

  Run the UpdateDetector API with your own regional *detector-id*.

  Alternatively, you can use the following AWS CLI command. Replace *12abc34d567e8fa901bc2d34e56789f0* with your own regional *detector-id*.

```
aws guardduty update-detector --detector-id 12abc34d567e8fa901bc2d34e56789f0 --
features '[{"Name" : "EKS_RUNTIME_MONITORING", "Status" : "DISABLED"}]'
```

- **To disable EKS Runtime Monitoring for member accounts in your organization**

  Run the UpdateMemberDetectors API with the regional *detector-id* of the delegated GuardDuty administrator account of the organization.

  Alternatively, you can use the following AWS CLI command. Replace *12abc34d567e8fa901bc2d34e56789f0* with the regional *detector-id* of the delegated GuardDuty administrator account of the organization and *111122223333* with the AWS account ID of the member account for which you want to disable this feature.

```
aws guardduty update-member-detectors --detector-id 12abc34d567e8fa901bc2d34e56789f0
 --account-ids 111122223333 --features '[{"Name" : "EKS_RUNTIME_MONITORING",
 "Status" : "DISABLED"}]'
```

- **To update EKS Runtime Monitoring auto-enable settings for your organization**

  Perform the following step only if you have configured the EKS Runtime Monitoring auto-enablement settings to either new (`NEW`) or all (`ALL`) member accounts in the organization. If you had already configured it as `NONE`, then you can skip this step.

> **Note**
>
> Setting the EKS Runtime Monitoring auto-enable configuration to `NONE` means that EKS Runtime Monitoring will not be enabled automatically for any existing member account or when a new member account joins your organization.

Run the UpdateOrganizationConfiguration API with the regional *detector-id* of the delegated GuardDuty administrator account of the organization.

Alternatively, you can use the following AWS CLI command. Replace *12abc34d567e8fa901bc2d34e56789f0* with the regional *detector-id* of the delegated GuardDuty administrator account of the organization. Replace the *EXISTING_VALUE* with your current configuration for auto-enabling GuardDuty.

```
aws guardduty update-organization-configuration --detector-
id 12abc34d567e8fa901bc2d34e56789f0 --auto-enable-organization-members EXISTING_VALUE
 --features '[{"Name" : "EKS_RUNTIME_MONITORING", "AutoEnable": "NONE"}]'
```

GuardDuty security agent release versions

GuardDuty releases an updated agent version from time to time. When GuardDuty manages the agent automatically, GuardDuty is designed to update the agent on your behalf. When you manage the agent manually, you are responsible to update the agent version for your resource types – Amazon EC2 instances, Amazon ECS clusters, and Amazon EKS clusters.

The following sections provide GuardDuty security agent release versions and associated release notes for all the supported resource types.

Topics

- GuardDuty security agent versions for Amazon EC2 instances
- GuardDuty security agent versions for AWS Fargate (Amazon ECS only)
- GuardDuty security agent versions for Amazon EKS resources
- Additional resources - next steps

GuardDuty security agent versions for Amazon EC2 instances

The following table shows the release version history for the GuardDuty security agent for Amazon EC2.

SSM distributor version	Agent version	Release notes	Availability date
v1.8.0	v1.8.0	General performance tuning and enhancements.	August 12, 2025
v1.7.2	v1.7.1	Improved support for local agent install for RPM based Linux distributions.	July 23, 2025
v1.7.1	1.7.1	Added support for Fedora 40 and Fedora 41. For a list of all verified OS distributions for Amazon EC2 resources, see [Validate architectural requirements](). General performance tuning and enhancements.	June 03, 2025
v1.7.0	v1.7.0	Added support for Oracle Linux versions 8.9 and 9.3, and Rocky Linux version 9.5. For a list of all verified OS distributions for Amazon EC2 resources, see	April 03, 2025

SSM distributor version	Agent version	Release notes	Availability date
		Validate architectural requirements. Improved container ID resolution. General performance tuning and enhancements.	
v1.6.0	v1.6.0	General performance tuning and enhancements.	February 6, 2025
v1.5.0	v1.5.0	Added support for CentOS Stream 9.0, RedHat 9.4, Fedora 34.0, and Ubuntu 24.04. Support for ARM instances for `.../MetadataDNSRebind` findings. General performance tuning and enhancements.	November 20, 2024
v1.3.1	v1.3.1	Support for custom DNS resolvers.	September 12, 2024

SSM distributor version	Agent version	Release notes	Availability date
v1.3.0	v1.3.0	General performance tuning and enhancements. Includes support to capture additional security signals for future GuardDuty Runtime Monitoring finding types.	August 19, 2024
v1.2.0	v1.2.0	Supports OS distributions Ubuntu 20.04, Ubuntu 22.04, Debian 11, and Debian 12. Supports kernel 6.5 and 6.8. General performance tuning and enhancements.	June 13, 2024

SSM distributor version	Agent version	Release notes	Availability date
v1.1.0	v1.1.0	Supports GuardDuty automated agent configuration in Runtime Monitorin g for Amazon EC2 instances. Supports new security signals and findings released with the announcement of general availability of Runtime Monitoring for EC2 instances. General performan ce tuning and enhancements.	March 26, 2024
v1.0.2	v1.0.2	Supports the latest Amazon ECS AMIs.	February 2, 2024
v1.0.1	v1.0.1	Agent versions released prior to v1.0.2 are incompati ble with Amazon ECS AMIs launched after January 31, 2024. General performan ce tuning and enhancements.	January 23, 2024

SSM distributor version	Agent version	Release notes	Availability date
v1.0.0	v1.0.0	Initial release of the RPM installation. Agent versions released prior to v1.0.2 are incompatible with Amazon ECS AMIs launched after January 31, 2024.	November 26, 2023

GuardDuty security agent versions for AWS Fargate (Amazon ECS only)

The following table shows the release version history for the GuardDuty security agent for Fargate (Amazon ECS only).

Agent version	Container image	Release notes	Availability date
v1.8.0	**x86_64 (AMD64)**: `sha256:4425417f39e38b24c1be428bacad1dad53e0645530dcf4422436353bfe358e3a` **x86_64 (AMD64)**: `sha256:aff069418fd6825846f8f575c49906a67c8a446d12d9ed0`	General performance tuning and enhancements.	August 12, 2025

Agent version	Container image	Release notes	Availability date
	`d21ab95bd 0d05497b`		
v1.7.0	**x86_64 (AMD64)**: `sha256:bf 9197abdf8 53607e5fa 392b4f97c cdd6ca56d d179be3ce 8849e552d 96582ac8` **Graviton (ARM64)**: `sha256:56 c8683c948 bcd82c0db cebf75520 4365ac728 5994693c1 1717bd45f 86e279c2`	Improved container ID resolution. General performan ce tuning and enhancements.	April 04, 2025

Agent version	Container image	Release notes	Availability date
v1.6.0	**x86_64 (AMD64)**: `sha256:c8dea71d372bc47b2f236f7a091b9a9b06bc8193c1cfe4c9346eb50f89258897` **Graviton (ARM64)**: `sha256:f4032a566b90537646c2a987bef42eca1b498078ccc58a848603f877971a8dbe`	General performance tuning and enhancements.	February 6, 2025

Agent version	Container image	Release notes	Availability date
v1.5.0	**x86_64 (AMD64)**: `sha256:5e6fdc41f9eb748219d0498cd6c1dba6a19d875daec50167a0ac80e5028eac54` **Graviton (ARM64)**: `sha256:d56801ff6864d6014740103b70b1c38431851358d182613bede20fe21090e734`	Support for ARM tasks for `.../MetadataDNSRebind` findings. General performance tuning and enhancements.	November 14, 2024

Agent version	Container image	Release notes	Availability date
v1.4.1	**x86_64 (AMD64)**: `sha256:ef36a11151ec2d3d7db22273bfb954750dee76f0ac7bec37a7ba7e74c3de1c78` **Graviton (ARM64)**: `sha256:a8844544a59d6b4cba98f8e528b513ac2d97432f208e3ad497cc16b331aa9faa`	Container image hardening. General performance tuning and enhancements.	October 24, 2024

Agent version	Container image	Release notes	Availability date
v1.3.1	**x86_64 (AMD64)**: `sha256:a6 e2307d796 e2875907b c4c1c6962 2c906f319 2ddc42ef2 7b99e0a8f 0979f3e0` **Graviton (ARM64)**: `sha256:ad 1b6539d80 6edb504f1 7e6bcfb8b 4026c5e82 2300afc31 c0d23c6a0 8f9b99e9`	Support for custom DNS resolvers.	September 11, 2024

Agent version	Container image	Release notes	Availability date
v1.3.0	**x86_64 (AMD64)**: `sha256:f1ad3fb2dc55a1110c60eecf4453b9f9c02f29acb261df39814e7d29296bf831` **Graviton (ARM64)**: `sha256:ff81a755d46681e409f55a95beedae9ebbcf5336e1c0b1e6348af7c6518bdbb1`	General performance tuning and enhancements. Includes support to capture additional security signals for future GuardDuty [GuardDuty Runtime Monitoring finding types]().	August 9, 2024

Agent version	Container image	Release notes	Availability date
v1.2.0	**x86_64 (AMD64)**: `sha256:1dbad20ac2dc66d52d00bb28dde4281fe0d3c5f261b1649b247c2369d9e26b93` **Graviton (ARM64)**: `sha256:91930f8446f5f95b93b8ccb18773992affa401eb3f42da89d68077a56bafa6cd`	General performance tuning and enhancements.	May 31, 2024

Agent version	Container image	Release notes	Availability date
v1.1.0	**x86_64 (AMD64)**: `sha256:83ce3cf2ef85a349ed1797a8cf30a008ac5d8c9f673f2835823957e9dcf71657` **Graviton (ARM64)**: `sha256:0d4b61648d7bdeab8ab8d94684f805498927c7d437d318204dcccfe8c9383dc7`	Supports new security signals and findings. General performance tuning and enhancements.	May 01, 2024

Agent version	Container image	Release notes	Availability date
v1.0.1	**x86_64 (AMD64)**: `sha256:9f8cd438fb66f62d09bfc641286439f7ed5177988a314a6021ef4ff880642e68` **Graviton (ARM64)**: `sha256:82c66bb615bd0d1e96db77b1f1fb51dc03220caa593b1962249571bf7147d1b7`	General performance tuning and enhancements.	January 26, 2024

Agent version	Container image	Release notes	Availability date
v1.0.0	**x86_64 (AMD64)**: `sha256:359b8b014e5076c625daa1056090e522631587a7afa3b2e055edda6bd1141017` **Graviton (ARM64)**: `sha256:b9438690fa8a86067180a11658bec0f4f838ae3fbd225d04b9306250648b3984`	Initial release of GuardDuty security agent for AWS Fargate (Amazon ECS only).	November 26, 2023

GuardDuty security agent versions for Amazon EKS resources

GuardDuty releases an updated agent version from time to time. When GuardDuty manages the agent automatically, it is designed to manage the agent updates on your behalf. When you manage the agent manually, you are responsible to update the agent version for your Amazon EKS clusters.

Before updating the agent to a specific version, add the image registry for GuardDuty to the `allowed-container-registries` in your admission controller. For more information, see Amazon ECR repository hosting GuardDuty agent.

The following table shows the release version history of Amazon EKS add-on GuardDuty agent.

Agent version	Container image	Release notes	Availability date	End of standard support[1]
v1.11.0	**x86_64 (AMD64)**: sha256:7c398fd50dee5fe493c361aa7fe191e094ddfa000c65b449a9ed6a5cd53610e7 **Graviton (ARM64)**: sha256:98a547b47d4770f45e75eb9730c807d9265722ca2f391e0aa5cb19e487ab455f	Added support for Fedora 40 and Fedora 41 OS distributions. For more information, see Validating architectural requirements (for EKS). General performance tuning and security enhancements.	August 29, 2025	–
v1.10.0	**x86_64 (AMD64)**: sha256:6dcbe5b055e1ef0af903071ede0b08f755ad5b7e9774a67df5399efdaa1f3d7d	Improved container ID resolution. General performance tuning and enhancements.	April 04, 2025	–

Agent version	Container image	Release notes	Availability date	End of standard support[1]
	Graviton (ARM64): `sha256:f0536882269610a4bab543abf93d3e070b1b559e62a2e67d82dfa9837600f72`			
v1.9.0	**x86_64 (AMD64)**: `sha256:51c5789ef6570f9bec879ac48a8f4769718cbc31e45430032569917e219af63f` **Graviton (ARM64)**: `sha256:9c2f74e7ea0827b7e422ae4c91fffc6c2bc41a1cdb96c7191d05259d337154e1`	General performance tuning and enhancements.	March 02, 2025	–

Agent version	Container image	Release notes	Availability date	End of standard support[1]
v1.8.1	**x86_64 (AMD64)**: `sha256:f2ce8cf89dbe17e3388cecb35053544dadf21af7770545f8d4b50384076aff47` **Graviton (ARM64)**: `sha256:30f586e4b694e704bcafadfa9081ab0aeff3cfbcde39743a0f1e24f77d79627f`	Added support for CentOS Stream 9.0, RedHat 9.4, Fedora 34.0, and Ubuntu 24.04. Support for ARM instances for `.../MetadataDNSRebind` finding. General performance tuning and enhancements.	November 23, 2024	–

Agent version	Container image	Release notes	Availability date	End of standard support[1]
v1.7.1	**x86_64 (AMD64)**: `sha256:b8b86b5d0872c8b67fecf64ec3d172666360545435a1752447d510951a7fd749` **Graviton (ARM64)**: `sha256:40ac4cfc354fd430ba7897ca1632e9a500ed13eeb0c315c5bcad38680e76b6e9`	General performance tuning and enhancements. Includes support to capture additional security signals for future GuardDuty Runtime Monitoring finding types. Support for custom DNS resolvers.	September 13, 2024	–

Agent version	Container image	Release notes	Availability date	End of standard support[1]
v1.7.0	**x86_64 (AMD64)**: `sha256:f3a2a8806e6c2a7fd63a91cccf6f7dffcd7e68554a423d610cea8c7e8f2185ec` **Graviton (ARM64)**: `sha256:b1a6db35a072c0de3c695e5e909a03e6c4e1fdbe47ecfaeb2784435cf67ebe0a`	General performance tuning and enhancements. Includes support to capture additional security signals for future GuardDuty Runtime Monitoring finding types.	August 17, 2024	–

Agent version	Container image	Release notes	Availability date	End of standard support[1]
v1.6.1	**x86_64 (AMD64)**: `sha256:30650708a6601f6d6b9046f54b30f5fd65af296b1e40b8c24426b9bdb07c3ab1` **Graviton (ARM64)**: `sha256:5f637c42ffb306b20f776d9d83e1e0b4be40ce245be44afcf43a8902b4d71019`	General performance tuning and enhancements.	May 14, 2024	–

Agent version	Container image	Release notes	Availability date	End of standard support[1]
v1.6.0	**x86_64 (AMD64)**: sha256:7dabcbee30d8b053676752fbc19e89f77272d9a6a53cc93731f5872180ef9010 **Graviton (ARM64)**: sha256:9710f53afccdf4f22b265a1a6fc27f1469403af1f7d5d08c4869a7269cdd2650	• Supports GuardDuty automated agent configuration for EKS/EC2 resources. • Supports the new security signals and findings. For more information, see Collected runtime event types that GuardDuty uses and GuardDuty Runtime Monitoring finding types. • General performance tuning and enhancements.	April 29, 2024	–

Agent version	Container image	Release notes	Availability date	End of standard support[1]
v1.5.0	**x86_64 (AMD64)**: `sha256:e09a4e70af4058a212f172cc8eb3fc23ad9bed547ed609faa2bb82cf7cc5532d` **Graviton (ARM64)**: `sha256:afc9a3f8f17ae12499d76069efcf1b46271a5a4b2b3f6ba5de54637b8f55d5c6`	• General performance tuning and enhancements. • Security enhancements including new event types under [Collected runtime event types](#). • Performance enhancements around CPU usage.	March 07, 2024	–

Agent version	Container image	Release notes	Availability date	End of standard support[1]
v1.4.1	**x86_64 (AMD64)**: `sha256:66d491927763742660faa87cc2c39bb97b7873039157ae8b90bc999cb73d0b9c` **Graviton (ARM64)**: `sha256:537a330b2dd82357024fb6daeb8761034b7defd43b10dffe0792c9e6d0778b40`	General performance tuning and enhancements.	January 16, 2024	–

Agent version	Container image	Release notes	Availability date	End of standard support[1]
v1.4.0	**x86_64 (AMD64)**: sha256:848ce13d9430bad554ac23d4699551505326ada2a88e1a721fe9f86b56b52c0f **Graviton (ARM64)**: sha256:0c650aeafeeb5f2bcb8b989ac849bedc1fae1a4de1cf6306ffdd9c6aebe67f8e	Manifest mount point support better data collection AppArmor configuration in manifest Collect command line argument General performance tuning and enhancements	December 21, 2023	–

Agent version	Container image	Release notes	Availability date	End of standard support[1]
v1.3.1	**x86_64 (AMD64)**: `sha256:55578fcb7b73097ade5c8404390ef16cf76a7b568490abaae01ac75992b3ea29` **Graviton (ARM64)**: `sha256:e3ce8d66ac2121f8d476eb58f8bc50ab51336647615eb7cf514c21421cb818fd`	Important security patches and updates.	October 23, 2023	–

Agent version	Container image	Release notes	Availability date	End of standard support[1]
v1.3.0	**x86_64 (AMD64)**: `sha256:6dace2337dfbb7609811be89fb4b23ae0b865f1027ad78fbe69530bfbd46c694` **Graviton (ARM64)**: `sha256:4928a7c6ef40e77c8ec95841323bb9a110db31f12c0ee7ab965e08b43efd01bb`	Supports Ubuntu platform Supports Kubernetes version 1.28 General performance enhancements and stability improvement.	October 05, 2023	–

Agent version	Container image	Release notes	Availability date	End of standard support[1]
v1.2.0	**x86_64 (AMD64)**: `sha256:d610413d662ec042057f05d6942496d7f2c08e9f5a077ea307ffdb5d3f11bcc3` **Graviton (ARM64)**: `sha256:174d7ab28b2f95e5309da80d95b88ad26f602dfe72c2b351a0ef9297a1412bfa`	In addition to AMD64-based instances, v1.2.0 now also supports ARM64-based instances. Added and verified support for Bottlerocket Supports Kubernetes version 1.27 General performance enhancements and stability improvements.	June 16, 2023	–

Agent version	Container image	Release notes	Availability date	End of standard support[1]
v1.1.0	sha256:b19ba3a3c1a508d153263ae2fda891a7928b5ca9b3a5692db6c101829303281c	In addition to Kubernetes versions supported by GuardDuty security agent, this agent release also supports Kubernetes version 1.26. General performance enhancements and stability improvements.	May 2, 2023	May 14, 2024
v1.0.0	sha256:e38bdd2b1323e89113f1a31bd4bc8e5a8098525dd98e6981a28b9906b1e4411e	Initial release of Amazon EKS add-on agent.	March 30, 2023	May 14, 2024

[1] For information about updating your current agent version that is approaching to an end of standard support, see Updating security agent manually for Amazon EKS resources.

Additional resources - next steps

For more information on the next steps, see the following topics:

- Prerequisites to enabling Runtime Monitoring - With new agent versions, there might be an update to the prerequisites section. Verify and validate that your resources meet the latest prerequisites.
- Managing GuardDuty security agents - When you manage the agent manually, then you're responsible for managing the updates to the agent version running on your resources. Based on your resource type (Amazon EKS or Amazon EC2-Amazon ECS), perform the steps to update the security agent. Also make sure to validate your VPC endpoint configuration.
- Reviewing runtime coverage statistics and troubleshooting issues - After you have updated the security agent, you can assess the runtime coverage your resource. If there is any coverage issue, then use the associated troubleshooting steps.

Disabling, uninstalling, and cleaning up resources in Runtime Monitoring

This section applies to your AWS account if you choose to disable Runtime Monitoring, or only GuardDuty automated agent configuration for a resource type.

Disabling GuardDuty automated agent configuration

GuardDuty doesn't remove the security agent that is deployed on your resource. However, GuardDuty will stop managing the updates to the security agent.

GuardDuty continues to receive the runtime events from your resource type. To prevent an impact on your usage statistics, make sure to remove the GuardDuty security agent from your resource.

Whether or not an AWS account uses a shared VPC endpoint, GuardDuty doesn't delete the VPC endpoint. If required, you will need to delete the VPC endpoint manually.

Disabling Runtime Monitoring and EKS Runtime Monitoring

This section applies to you in the following scenarios:

- You never enabled EKS Runtime Monitoring separately and now you disabled Runtime Monitoring.
- You are disabling both Runtime Monitoring and EKS Runtime Monitoring. If you're unsure about the configuration status of EKS Runtime Monitoring, see Checking EKS Runtime Monitoring configuration status.

> **Disabling Runtime Monitoring without disabling EKS Runtime Monitoring**
>
> In this scenario, at some point in time, you enabled EKS Runtime Monitoring, and later, also enabled Runtime Monitoring without disabling EKS Runtime Monitoring. Now, when you disable Runtime Monitoring, you will also need to disable EKS Runtime Monitoring; otherwise, you will continue incurring usage cost for EKS Runtime Monitoring.

If the previously listed scenarios apply to you, then GuardDuty will take the following actions in your account:

- GuardDuty deletes the VPC endpoint that has the `GuardDutyManaged:true` tag. This is the VPC that GuardDuty had created to manage the automated security agent.
- GuardDuty deletes the security group that was tagged as `GuardDutyManaged:true`.
- For a shared VPC that has been used by at least one participant account, GuardDuty neither deletes the VPC endpoint nor the security group associated with the shared VPC resource.
- For an Amazon EKS resource, GuardDuty deletes the security agent. This is independent of whether it managed manually or through GuardDuty.

  For an Amazon ECS resource, because an ECS task is immutable, GuardDuty can't uninstall the security agent from that resource. This is independent of how you manage the security agent – manually or automatically through GuardDuty. After you disable Runtime Monitoring, GuardDuty will not attach a sidecar container when a new ECS task starts running. For information about working with Fargate-ECS tasks, see How Runtime Monitoring works with Fargate (Amazon ECS only).

  For an Amazon EC2 resource, GuardDuty uninstalls the security agent from all the Systems Manager (SSM) managed Amazon EC2 instances only when it meets the following conditions:

  - Your resource is **not** tagged with `GuardDutyManaged:false` exclusion tag.
  - GuardDuty must have permissions to access the tags in instance metadata. For this EC2 resource, the **Access to tags in instance metadata** is set to **Allow**.

When you stop managing the security agent manually

Regardless of which approach you use to deploy and manage the GuardDuty security agent, to stop monitoring the runtime events in your resource, you must remove the GuardDuty security agent. When you want to stop monitoring the runtime events from a resource type in an account, you may also delete the Amazon VPC endpoint.

Uninstalling security agent manually for Amazon EC2 resources

This section provides methods to uninstall the GuardDuty security agent from your Amazon EC2 resources. When you manage the security agent manually, you're responsible to remove the agent from the resources. GuardDuty will not take any action on the resources that you manage.

If you created an Amazon VPC endpoint manually, then after you uninstall the security agent on all the monitored resource types in your account, you can choose to delete the VPC endpoint. This is a separate step. For more information, see To delete a VPC endpoint.

Based on how you installed the security agent in your resource, choose one of the following methods to uninstall it.

Topics

- Method 1 - By using the Run command
- Method 2 - By using Linux Package Managers

Method 1 - By using the Run command

When you installed the security agent with Method 1 - Using AWS Systems Manager, perform the following steps to uninstall the agent:

To uninstall the GuardDuty security agent

1. You can uninstall the GuardDuty security agent by following the steps as specified in AWS Systems Manager Run Command in the *AWS Systems Manager User Guide*. Use the Uninstall action in the parameters to uninstall the GuardDuty security agent.

   In the **Targets** section, make sure that the impact is only on those Amazon EC2 instances from which you want to uninstall the security agent.

   Use the following GuardDuty document and distributor:

   - Document name: `AmazonGuardDuty-ConfigureRuntimeMonitoringSsmPlugin`
   - Distributor: `AmazonGuardDuty-RuntimeMonitoringSsmPlugin`

2. After providing all the details, when you choose **Run**, the security agent that it deployed on the targeted Amazon EC2 instances is removed.

   To remove the Amazon VPC endpoint configuration, you must disable both Runtime Monitoring and Amazon EKS Runtime Monitoring.

3. If you also want to delete the VPC endpoint that is associated with this security agent, then see To delete a VPC endpoint.

Method 2 - By using Linux Package Managers

When you installed the security agent with Method 2 - Using Linux Package Managers, perform the following steps to uninstall the agent:

To uninstall the GuardDuty security agent

1. Connect to the your instance. For steps on how to do this, see Connect to your Linux instance using an SSH client in the *Amazon EC2 User Guide*.

2. **Command to uninstall**

   The following command will uninstall the GuardDuty security agent from the Amazon EC2 instance to which you connect:

   - For RPM:

     ```
     sudo rpm -e amazon-guardduty-agent
     ```

   - For Debian:

     ```
     sudo dpkg --purge amazon-guardduty-agent
     ```

   After you run the command, you can also check the logs associated with the command.

3. If you also want to delete the VPC endpoint that is associated with this security agent, then see To delete a VPC endpoint.

Cleaning up security agent resources

This section explains how you can clean up the AWS resources associated with the security agent. As listed in Disabling, uninstalling, and resource cleanup, GuardDuty will not delete or remove all the security agent resources. The following section provides instructions on how you can delete the security agent resources.

To delete Amazon VPC endpoint

When you manage the security agent manually, you may have created an Amazon VPC endpoint manually. After uninstalling the security agent for all the monitored resources in your account, you can choose to delete this VPC endpoint.

The following list provides scenarios when using a shared VPC compared to not using a shared VPC.

- Without a shared VPC – When you no longer want to monitor a resource in an account, consider deleting the Amazon VPC endpoint.
- With a shared VPC – When a shared VPC owner account deletes the shared VPC resource that was still being used, the Runtime Monitoring (and when applicable, EKS Runtime Monitoring) coverage status for the resources in your shared VPC owner account and the participating account might become unhealthy. For information about coverage status, see Reviewing runtime coverage statistics and troubleshooting issues.

For deleting the VPC endpoint, see Delete an interface endpoint in the *AWS PrivateLink Guide*.

To delete the security group

- Without a shared VPC – When you no longer want to monitor a resource type in an account, consider deleting the security group associated with the Amazon VPC.
- With a shared VPC – When the shared VPC owner account deletes the security group, any participant account that is currently using the security group associated with the shared VPC, the Runtime Monitoring coverage status for the resources in your shared VPC owner account and the participating account might become unhealthy. For more information, see Reviewing runtime coverage statistics and troubleshooting issues.

For information about steps, see Delete an Amazon EC2 security group in the *Amazon EC2 User Guide*.

To remove GuardDuty security agent from an EKS cluster

To remove the security agent from your EKS cluster that you no longer want to monitor, see Removing an Amazon EKS add-on from a cluster in the *Amazon EKS User Guide*.

Removing the EKS add-on agent doesn't remove the `amazon-guardduty` namespace from the EKS cluster. To delete the `amazon-guardduty` namespace, see Deleting a namespace.

To delete the `amazon-guardduty` namespace (EKS cluster)

Disabling Automated agent configuration doesn't automatically remove the `amazon-guardduty` namespace from your EKS cluster. To delete the `amazon-guardduty` namespace, see Deleting a namespace.

GuardDuty Malware Protection for EC2

Malware Protection for EC2 helps you detect the potential presence of malware by scanning the Amazon Elastic Block Store (Amazon EBS) volumes that are attached to Amazon Elastic Compute Cloud (Amazon EC2) instances and container workloads running on Amazon EC2. Malware Protection for EC2 provides scan options where you can decide if you want to include or exclude specific Amazon EC2 instances at the time of scanning. It also provides an option to retain the snapshots of Amazon EBS volumes attached to the Amazon EC2 instances or container workloads, in your GuardDuty accounts. The snapshots get retained only when malware is found and Malware Protection for EC2 findings are generated.

Malware Protection for EC2 is designed in a way that it won't affect the performance of your resources. For information about how Malware Protection for EC2 works within GuardDuty, see How GuardDuty scans EBS volumes for malware detection. For information about availability of Malware Protection for EC2 in different AWS Regions, see Regions and endpoints.

Notes

Malware Protection for EC2 **supports** malware scans on managed instances for Amazon EKS Auto Mode.

Malware Protection for EC2 **doesn't support** malware scans for AWS Fargate workloads running on with either Amazon EKS or Amazon ECS.

For information about these Amazon EKS features, see What is Amazon EKS? in the Amazon EKS User Guide.

Topics

- Comparing GuardDuty-initiated malware scan and On-demand malware scan
- How GuardDuty scans EBS volumes for malware detection
- Supported Amazon EBS volumes for malware scan
- Set up snapshot retention and EC2 scan coverage
- GuardDuty-initiated malware scan
- On-demand malware scan in GuardDuty
- Monitoring scan statuses and results in Malware Protection for EC2
- GuardDuty service accounts by AWS Region

- Quotas in Malware Protection for EC2

Comparing GuardDuty-initiated malware scan and On-demand malware scan

Malware Protection for EC2 offers two types of scans to detect potentially malicious activity in your Amazon EC2 instances and container workloads – GuardDuty-initiated malware scan and On-demand malware scan. The following table shows the comparison between both the scan types.

Factor	GuardDuty-initiated malware scan	On-demand malware scan
How the scan gets invoked	After you enable GuardDuty -initiated malware scan, whenever GuardDuty generates a finding that indicates the potential presence of malware in an Amazon EC2 instance or a container workload, GuardDuty automatic ally initiates an agentless malware scan on the Amazon EBS volumes attached to your potentially impacted resource. For more informati on, see GuardDuty-initiated malware scan.	You can initiate an On-demand malware scan by providing the Amazon Resource Name (ARN) of your Amazon EC2 instance. You can initiate an On-demand malware scan even when no GuardDuty finding is generated for your resource. For more information, see On-demand malware scan in GuardDuty.
Configuration needed	To use GuardDuty-initiate d malware scan, you must enable it for your account. To manage multiple accounts by using AWS Organizations or invitation based method, see Enabling GuardDuty-	Your account must have GuardDuty enabled. To use On-demand malware scan, there is no configuration required at the feature-level.

Factor	GuardDuty-initiated malware scan	On-demand malware scan
	initiated malware scan in multiple-account environments. To enable GuardDuty -initiated malware scan in your own account, see Enabling GuardDuty-initiated malware scan for a standalone account.	
Wait time to initiate a new scan	Whenever GuardDuty generates one of the Findings that invoke GuardDuty-initiated malware scan, a malware scan initiates automatically only once every 24 hours.	You can initiate an On-demand malware scan on the same resource any time after 1 hour from the start time of the previous scan.
Availability of the 30-day free trial period [1]	When you enable GuardDuty -initiated malware scan for the first time in your account, you can use a 30-day free trial period. For more information, see 30-day free trial in GuardDuty-initiated malware scan.	There is no free trial period with On-demand malware scan for new or existing GuardDuty accounts.

Factor	GuardDuty-initiated malware scan	On-demand malware scan
Scan options[2]	After you've configured GuardDuty-initiated malware scan, Malware Protection for EC2 provides the option to scan or skip specific Amazon EC2 resources by using tags. Malware Protection for EC2 will not initiate an automatic scan on the resources that you choose to exclude from scanning. For more informati on, see Scan options with user-defined tags.	Because you provide the resource ARN to start an on-demand malware scan manually, using Scan options with user-defined tags is not applicable.

[1]You will incur usage cost for creating EBS volume snapshots and retaining snapshots. For more information about configuring your account to retain snapshots, see Snapshots retention.

[2]Both GuardDuty-initiated malware scan and On-demand malware scan support using a global tag to exclude Amazon EC2 resources from malware scans. For more information, see Global `GuardDutyExcluded` tag.

How GuardDuty scans EBS volumes for malware detection

This section explains how Malware Protection for EC2, including both GuardDuty-initiated malware scan and On-demand malware scan, scans the Amazon EBS volumes associated with your Amazon EC2 instances and container workloads. Before proceeding, consider the following customizations:

- Scan options – Malware Protection for EC2 offers the capability to specify tags to either include or exclude Amazon EC2 instances and Amazon EBS volumes from the scanning process. Only GuardDuty-initiated malware scan supports scan options with user-defined tags. Both GuardDuty-initiated malware scan and On-demand malware scan support the global `GuardDutyExcluded` tag. For more information, see Scan options with user-defined tags.

- Snapshots retention – Malware Protection for EC2 provides an option to retain the snapshots of your Amazon EBS volumes in your AWS account. By default, this setting is turned off. You can opt in for snapshots retention for both GuardDuty initiated and on-demand malware scans. For more information, see Snapshots retention.

When GuardDuty generates one or more Findings that invoke GuardDuty-initiated malware scan, then this activity will be a reason for GuardDuty to initiate a malware scan. If your scan options do not exclude this instance, then GuardDuty will initiate the scan.

To initiate an On-demand malware scan on the Amazon EBS volumes associated with an Amazon EC2 instance, provide the Amazon Resource Name (ARN) of the Amazon EC2 instance.

As a response to starting an on-demand malware scan or an automatic GuardDuty-initiated malware scan, GuardDuty creates snapshots of the relevant EBS volumes attached to the potentially impacted resource, and shares them with the GuardDuty service account. When GuardDuty creates snapshot of your EBS volumes, it adds a default tag called `GuardDutyScanId`. This tag helps GuardDuty to access the snapshot. Make sure that you don't remove this tag. From these snapshots, GuardDuty creates an encrypted replica EBS volume in the service account.

After the scan completes, GuardDuty deletes the encrypted replica EBS volumes and the snapshots of your EBS volumes. By default, snapshots retention setting is turned off. However, snapshots are retained if Amazon EBS snapshot locking is enabled for them, regardless of the scan results and settings. GuardDuty can't modify the Amazon EBS snapshot lock settings.

The following list describes snapshots retention behavior, regardless of EBS snapshot locking:

Snapshots retention is turned on:

- When malware is found, GuardDuty retains the snapshots in your AWS account.
- When no malware is found, GuardDuty **doesn't** retain the snapshots unless they are locked.

Snapshots retention is turned off (default setting):

- Whether or not malware is found, the snapshots are not retained.
- GuardDuty can't delete locked Amazon EBS snapshots.

GuardDuty will retain each replica EBS volume in the service account for up to 55 hours. If there is a service outage, or failure with a replica EBS volume and its malware scan, GuardDuty will retain such an EBS volume for no more than seven days. The extended volume retention period is to triage and address the outage or failure. GuardDuty Malware Protection for EC2 will delete the replica EBS volumes from the service account after the outage or failure is addressed, or once the extended retention period lapses.

For information about GuardDuty malware detection methodology and the scan engines that it uses, see GuardDuty malware detection scan engine.

Supported Amazon EBS volumes for malware scan

In all of the AWS Regions where GuardDuty supports the Malware Protection for EC2 feature, you can scan the Amazon EBS volumes that are unencrypted or encrypted. You can have Amazon EBS volumes that are encrypted with either AWS managed key or customer managed key. Presently, some of the Regions where Malware Protection for EC2 is available, may support both the ways to encrypt your Amazon EBS volumes, while others support only customer managed key. For information about supported Regions, see and GuardDuty service accounts by AWS Region. For information about Regions where GuardDuty is available but Malware Protection for EC2 is not available, see Region-specific feature availability.

The following list describes the key that GuardDuty uses whether or not your Amazon EBS volumes are encrypted:

- **Amazon EBS volumes that are either unencrypted or encrypted with AWS managed key** – GuardDuty uses its own key to encrypt the replica Amazon EBS volumes.

  If your Region doesn't support scanning Amazon EBS volumes that are encrypted with Amazon EBS encryption by default, then you need to modify the default key to be a customer managed key. This will help GuardDuty access these EBS volumes. By modifying the key, even the future EBS volumes will get created with the updated key so that GuardDuty can support malware scans. For steps to modify the default key, see Modify default AWS KMS key ID of an Amazon EBS volume in next section.
- **Amazon EBS volumes that are encrypted with customer managed key** – GuardDuty uses the same key to encrypt the replica EBS volume. For information about what AWS KMS encryption related policies are supported, see Service-linked role permissions for Malware Protection for EC2.

Modify default AWS KMS key ID of an Amazon EBS volume

When you use create an Amazon EBS volume by using Amazon EBS encryption, and do not specify AWS KMS key ID, your Amazon EBS volume gets encrypted with a default key for encryption.

When you enable encryption by default, Amazon EBS will automatically encrypt new volumes and snapshots by using your default KMS key for Amazon EBS encryption.

You can modify the default encryption key and use a customer managed key for Amazon EBS encryption. This will help GuardDuty access these Amazon EBS volumes. To modify the EBS default key ID, add the following necessary permission to your IAM policy – `ec2:modifyEbsDefaultKmsKeyId`. Any newly-created Amazon EBS volume that you choose to be encrypted but don't specify an associated KMS key ID, will use the default key ID. Use one of the following methods to update the EBS default key ID:

To modify default KMS key ID of an Amazon EBS volume

Do one of the following:

- **Using an API** – You can use the ModifyEbsDefaultKmsKeyId API. For information about how you can view the encryption status of your volume, see Create Amazon EBS volume.
- **Using AWS CLI command** – The following example modifies the default KMS key ID that will encrypt Amazon EBS volumes if you don't provide a KMS key ID. Make sure to replace the Region with the AWS Region of your KM key ID.

  ```
  aws ec2 modify-ebs-default-kms-key-id --region us-west-2 --kms-key-
  id AKIAIOSFODNN7EXAMPLE
  ```

  The above command will generate an output similar to the following output:

  ```
  {
    "KmsKeyId": "arn:aws:kms:us-west-2:444455556666:key/AKIAIOSFODNN7EXAMPLE"
  }
  ```

  For more information, see modify-ebs-default-kms-key-id.

Set up snapshot retention and EC2 scan coverage

This section explains how to customize malware scanning options for your Amazon EC2 instances. These customizations apply to both On-demand malware scan and those initiated by GuardDuty. You can do the following:

- Enable snapshot retention – When enabled before a scan, GuardDuty will retain the Amazon EBS snapshot that GuardDuty detected as malicious.

- Choose which Amazon EC2 instances to scan – Use tags to include or exclude specific Amazon EC2 instances from malware scans.

Snapshots retention

GuardDuty provides you with the option to retain the snapshots of your EBS volumes in your AWS account. By default, the snapshots retention setting is turned off. The snapshots will only be retained if you have this setting turned on before the scan initiates.

As the scan initiates, GuardDuty generates the replica EBS volumes based on the snapshots of your EBS volumes. After the scan completes and the snapshots retention setting in your account was turned on already, the snapshots of your EBS volumes will be retained only when malware is found and Malware Protection for EC2 finding types get generated. When no malware is found, then regardless of your snapshot settings, GuardDuty automatically deletes the snapshots of your EBS volumes unless Amazon EBS snapshot locking has been enabled on the created snapshots.

Snapshots usage cost

During the malware scanning, as GuardDuty creates the snapshots of your Amazon EBS volumes, there is a usage cost associated with this step. If you turn on the snapshots retention setting for your account, when malware is found and the snapshots get retained, you will incur usage cost for the same. For information about cost of snapshots and their retention, see Amazon EBS pricing.

As a delegated GuardDuty administrator account, only you can make this update on behalf of the organization member accounts. However, if a member account is managed by invitation method, they can make this change on their own. For more information, see Administrator account and member account relationships.

Choose your preferred access method to turn on the snapshots retention setting.

Console

1. Open the GuardDuty console at https://console.aws.amazon.com/guardduty/.
2. In the navigation pane, under **Protection plans**, choose **Malware Protection for EC2**.
3. Choose **General settings** in the bottom section of the console. To retain the snapshots, turn on **Snapshots retention**.

API/CLI

Run UpdateMalwareScanSettings to update the current configuration for snapshot retention setting.

Alternatively, you can run the following AWS CLI command to automatically retain snapshots when GuardDuty Malware Protection for EC2 generates findings.

Ensure to replace the *detector-id* with your own valid detectorId.

To find the detectorId for your account and current Region, see the **Settings** page in the https://console.aws.amazon.com/guardduty/ console, or run the ListDetectors API.

```
aws guardduty update-malware-scan-settings --detector-
id 60b8777933648562554d637e0e4bb3b2 --ebs-snapshot-preservation
 "RETENTION_WITH_FINDING"
```

If you want to turn off snapshots retention, replace RETENTION_WITH_FINDING with NO_RETENTION.

Scan options with user-defined tags

By using GuardDuty-initiated malware scan, you can also specify tags to either include or exclude Amazon EC2 instances and Amazon EBS volumes from the scanning and threat detection process. You can customize each GuardDuty-initiated malware scan by editing tags in either the inclusion or exclusion tags list. Each list can include up to 50 tags.

If you don't already have user-defined tags associated to your EC2 resources, see Tag your Amazon EC2 resources in the *Amazon EC2 User Guide*.

Note

On-demand malware scan doesn't support scan options with user-defined tags. It supports Global GuardDutyExcluded tag.

To exclude EC2 instances from malware scan

If you want to exclude any Amazon EC2 instance or Amazon EBS volume during the scanning process, you can set the GuardDutyExcluded tag to true for any Amazon EC2 instance or Amazon EBS volume, and GuardDuty won't scan it. For more information about `GuardDutyExcluded` tag, see Service-linked role permissions for Malware Protection for EC2. You can also add an Amazon EC2 instance tag to an exclusion list. If you add multiple tags to the exclusion tags list, any Amazon EC2 instance that contains at least one of these tags will be excluded from the malware scanning process.

As a delegated GuardDuty administrator account, only you can make this update on behalf of the organization member accounts. However, if a member account is managed by invitation method, they can make this change on their own. For more information, see Administrator account and member account relationships.

Choose your preferred access method to add a tag associated with an Amazon EC2 instance, to an exclusion list.

Console

1. Open the GuardDuty console at https://console.aws.amazon.com/guardduty/.
2. In the navigation pane, under **Protection plans**, choose **Malware Protection for EC2**.
3. Expand **Inclusion/Exclusion tags** section. Choose **Add tags**.
4. Choose **Exclusion tags** and then choose to **Confirm**.
5. Specify the tag's **Key** and **`Value`** pair that you want to exclude. It is optional to provide the **`Value`**. After you add all the tags, choose **Save**.

   > **Important**
   >
   > Tag keys and values are case-sensitive. For more information, see Tag restrictions in the *Amazon EC2 User Guide*.

   If a value for a key is not provided and the EC2 instance is tagged with the specified key, this EC2 instance will be excluded from the GuardDuty-initiated malware scan scanning process, regardless of the tag's assigned value.

API/CLI

Run UpdateMalwareScanSettings by excluding an EC2 instance or a container workload from the scanning process.

The following AWS CLI example command adds a new tag to the exclusion tags list. Replace the example *detector-id* with your own valid `detectorId`.

`MapEquals` is a list of `Key/Value` pairs.

To find the `detectorId` for your account and current Region, see the **Settings** page in the https://console.aws.amazon.com/guardduty/ console, or run the ListDetectors API.

```
aws guardduty update-malware-scan-settings --detector-
id 60b8777933648562554d637e0e4bb3b2 --scan-resource-criteria '{"Exclude":
 {"EC2_INSTANCE_TAG" : {"MapEquals": [{ "Key": "TestKeyWithValue", "Value":
 "TestValue" }, {"Key":"TestKeyWithoutValue"} ]}}}' --ebs-snapshot-preservation
 "RETENTION_WITH_FINDING"
```

Important

Tag keys and values are case-sensitive. For more information, see Tag restrictions in the *Amazon EC2 User Guide*.

To include EC2 instances in malware scan

If you want to scan an EC2 instance, add its tag to the inclusion list. When you add a tag to an inclusion tags list, an EC2 instance that doesn't contain any of the added tags is skipped from the malware scan. If you add multiple tags to the inclusion tags list, an EC2 instance that contains at least one of those tags is included in the malware scan. Sometimes, an EC2 instance may be skipped during the scanning process because of other reasons. For more information, see Reasons for skipping resource during malware scan.

As a delegated GuardDuty administrator account, only you can make this update on behalf of the organization member accounts. However, if a member account is managed by invitation method, they can make this change on their own. For more information, see Administrator account and member account relationships.

Choose your preferred access method to add a tag associated with an EC2 instance, to an inclusion list.

Console

1. Open the GuardDuty console at https://console.aws.amazon.com/guardduty/.

2. In the navigation pane, under **Protection plans**, choose **Malware Protection for EC2**.
3. Expand **Inclusion/Exclusion tags** section. Choose **Add tags**.
4. Choose **Inclusion tags** and then choose **Confirm**.
5. Choose **Add new inclusion tag** and specify the tag's **Key** and **`Value`** pair that you want to include. It is optional to provide the **`Value`**.

   After you have added all the inclusion tags, choose **Save**.

   If a value for a key is not provided an EC2 instance is tagged with the specified key, the EC2 instance will be included in the Malware Protection for EC2 scanning process, regardless of the tag's assigned value.

API/CLI

- Run UpdateMalwareScanSettings to include an EC2 instance or a container workload in the scanning process.

  The following AWS CLI example command adds a new tag to the inclusion tags list. Ensure that you replace the example *`detector-id`* with your own valid `detectorId`. Replace the example *`TestKey`* and *`TestValue`* with the `Key` and `Value` pair of the tag associated with your EC2 resource.

  `MapEquals` is a list of `Key`/`Value` pairs.

  To find the `detectorId` for your account and current Region, see the **Settings** page in the https://console.aws.amazon.com/guardduty/ console, or run the ListDetectors API.

```
aws guardduty update-malware-scan-settings --detector-
id 60b8777933648562554d637e0e4bb3b2 --scan-resource-criteria '{"Include":
 {"EC2_INSTANCE_TAG" : {"MapEquals": [{ "Key": "TestKeyWithValue", "Value":
 "TestValue" }, {"Key":"TestKeyWithoutValue"} ]}}}' --ebs-snapshot-preservation
 "RETENTION_WITH_FINDING"
```

  Important

  Tag keys and values are case-sensitive. For more information, see Tag restrictions in the *Amazon EC2 User Guide*.

> **Note**
>
> It may take up to 5 minutes for GuardDuty to detect a new tag.

At any time, you can either choose **Inclusion tags** or **Exclusion tags** but not both. If you want to switch between the tags, choose that tag from the dropdown menu when you add new tags, and **Confirm** your selection. This action clears all your current tags.

Global `GuardDutyExcluded` tag

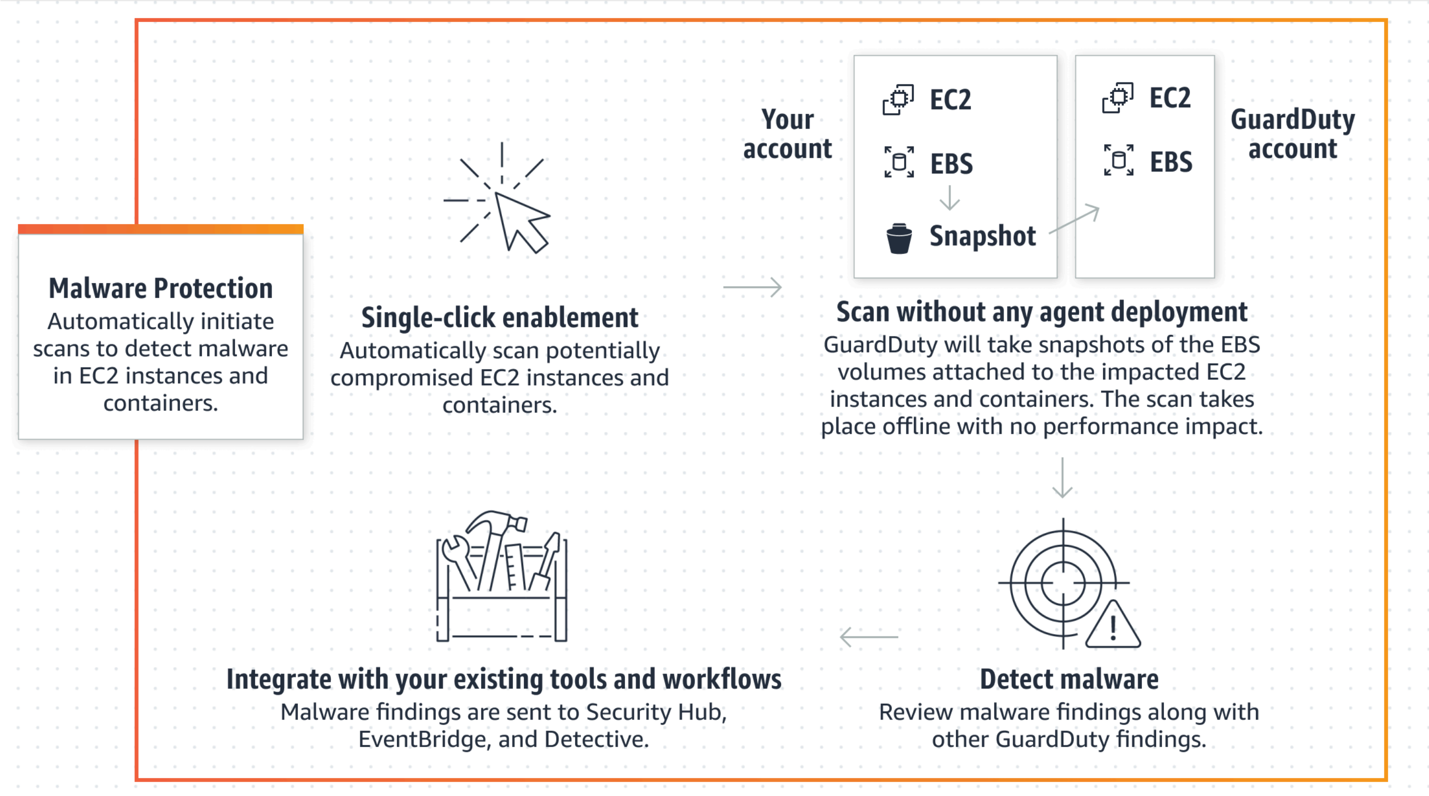
GuardDuty uses a global tag key, `GuardDutyExcluded`, that you can add to your Amazon EC2 resources and set the tag value to `true`. This Amazon EC2 resource that has this tag key and value pair will be excluded from the malware scan. Both the scan types (GuardDuty-initiated malware scan and On-demand malware scan) support the global tag. If you start an on-demand malware scan on an Amazon EC2, a scan ID will be generated. However, the scan will be skipped with an `EXCLUDED_BY_SCAN_SETTINGS` reason. For more information, see [Reasons for skipping resource during malware scan](#).

GuardDuty-initiated malware scan

With GuardDuty-initiated malware scan enabled, whenever GuardDuty generates [Findings that invoke GuardDuty-initiated malware scan](#), an agentless malware scan on the Amazon Elastic Block Store (Amazon EBS) volumes attached to the potentially impacted Amazon EC2 resource will initiate. Before a scan initiates, you must prepare your account for any customizations. With scan options, you can add inclusion tags associated with the resources that you want to scan, or add exclusion tags associated with the resources that you want to skip from the scanning process. An automatic scan initiation will always consider your scan options. GuardDuty also supports a global `GuardDutyExcluded:true` tag key:value pair. When you add this global tag to an Amazon EC2 resource, GuardDuty will initiate the scan and then skip it. You can also choose to turn on the snapshots retention setting to retain the snapshots of your EBS volumes where malware was potentially detected. For more information about scan options, global exclusion tag, and snapshot settings, see [Set up snapshot retention and EC2 scan coverage](#).

When GuardDuty generates multiple findings for the same Amazon EC2 resource, GuardDuty will be able to initiate a scan only after 24 hours have been passed since the last GuardDuty-initiated malware scan. For information about how the Amazon EBS volumes attached to your Amazon EC2 instance or container workload are scanned, see [How GuardDuty scans EBS volumes for malware detection](#).

The following image describes how GuardDuty-initiated malware scan works.



For information about GuardDuty malware detection methodology and the scan engines that it uses, see [GuardDuty malware detection scan engine](#).

When malware is found, GuardDuty generates [Malware Protection for EC2 finding types](#). If GuardDuty doesn't generate a finding indicative of malware on the same resource, no GuardDuty-initiated malware scan will be invoked. You can also initiate an On-demand malware scan on the same resource. For more information, see [On-demand malware scan in GuardDuty](#).

30-day free trial in GuardDuty-initiated malware scan

You can choose to enable or disable GuardDuty-initiated malware scan for an AWS account in a supported AWS Region at any time. If you have an organization, each member account has its own 30-day free trial.

To understand how 30-day free trial works, consider the following scenarios:

- When you enable GuardDuty for the first time (new GuardDuty account), GuardDuty-initiated malware scan also gets enabled and is included in the 30-day free trial associated with the GuardDuty service.
- An existing GuardDuty account can enable GuardDuty-initiated malware scan for the first time with a 30-day free trial. When you enable this feature in a different Region for the first time, you will get a 30-day free trial in that Region.
- If you have been using Malware Protection for EC2 in an AWS Region before this protection plan was divided into two scan types – GuardDuty-initiated malware scan and On-demand malware scan, you can continue using GuardDuty-initiated malware scan with the same pricing model in the same AWS Region. If you enable GuardDuty-initiated malware scan for the first time in a new Region, your account will get a 30-day free trial.

> **Note**
>
> Even if you're on a 30-day free trial period, the standard usage cost for creating the Amazon EBS volume snapshots and their retention applies. For more information, see Amazon EBS pricing.

Enabling GuardDuty-initiated malware scan in multiple-account environments

In a multiple-account environment, only GuardDuty administrator account can enable GuardDuty-initiated malware scan on behalf of their member accounts. Additionally, an administrator account that manages the member accounts with AWS Organizations support can choose to have GuardDuty-initiated malware scan enabled automatically on all the existing and new accounts in the organization. For more information, see Managing GuardDuty accounts with AWS Organizations.

Establishing trusted access to enable GuardDuty-initiated malware scan

If the GuardDuty delegated administrator account is not the same as the management account in your organization, the management account must enable GuardDuty-initiated malware scan for their organization. This way, the delegated administrator account can create the Service-linked role permissions for Malware Protection for EC2 in member accounts that are managed through AWS Organizations.

> **Note**
>
> Before you designate a delegated GuardDuty administrator account, see Considerations and recommendations.

Choose your preferred access method to allow the delegated GuardDuty administrator account to enable GuardDuty-initiated malware scan for member accounts in the organization.

Console

1. Open the GuardDuty console at https://console.aws.amazon.com/guardduty/.

   To log in, use the management account for your AWS Organizations organization.
2. a. If you have not designated a delegated GuardDuty administrator account, then:

      On the **Settings** page, under **delegated GuardDuty administrator account**, enter the 12-digit **`account ID`** that you want to designate to administer the GuardDuty policy in your organization. Choose **Delegate**.

   b. i. If you've already designated a delegated GuardDuty administrator account that is different from the management account, then:

         On the **Settings** page, under **Delegated Administrator**, turn on the **Permissions** setting. This action will allow the delegated GuardDuty administrator account to attach relevant permissions to the member accounts and enable GuardDuty-initiated malware scan in these member accounts.

      ii. If you've already designated a delegated GuardDuty administrator account that is the same as the management account, then you can directly enable GuardDuty-initiated malware scan for the member accounts. For more information, see Auto-enable GuardDuty-initiated malware scan for all member accounts.

   > **Tip**
   >
   > If the delegated GuardDuty administrator account is different from your management account, you must provide permissions to the delegated GuardDuty administrator account to allow enabling GuardDuty-initiated malware scan for member accounts.

3. If you want to allow the delegated GuardDuty administrator account to enable GuardDuty-initiated malware scan for member accounts in other Regions, change your AWS Region, and repeat the steps above.

API/CLI

1. Using your management account credentials, run the following command:

```
aws organizations enable-aws-service-access --service-principal malware-
protection.guardduty.amazonaws.com
```

2. (Optional) to enable GuardDuty-initiated malware scan for the management account that is not a delegated administrator account, the management account will first create the Service-linked role permissions for Malware Protection for EC2 explicitly in their account, and then enable GuardDuty-initiated malware scan from the delegated administrator account, similar to any other member account.

```
aws iam create-service-linked-role --aws-service-name malware-
protection.guardduty.amazonaws.com
```

3. You have designated the delegated GuardDuty administrator account in the currently selected AWS Region. If you have designated an account as a delegated GuardDuty administrator account in one region, that account must be your delegated GuardDuty administrator account in all other regions. Repeat the step above for all other Regions.

Configuring GuardDuty-initiated malware scan for delegated GuardDuty administrator account

Choose your preferred access method to enable or disable GuardDuty-initiated malware scan for a delegated GuardDuty administrator account.

Console

1. Open the GuardDuty console at https://console.aws.amazon.com/guardduty/.
2. In the navigation pane, choose **Malware Protection for EC2**.
3. On the **Malware Protection for EC2** page, choose **Edit** next to **GuardDuty-initiated malware scan**.
4. Do one of the following:

Using Enable for all accounts

- Choose **Enable for all accounts**. This will enable the protection plan for all the active GuardDuty accounts in your AWS organization, including the new accounts that join the organization.
- Choose **Save**.

Using Configure accounts manually

- To enable the protection plan only for the delegated GuardDuty administrator account account, choose **Configure accounts manually**.
- Choose **Enable** under the **delegated GuardDuty administrator account (this account)** section.
- Choose **Save**.

API/CLI

Run the [updateDetector](#) API operation using your own regional detector ID and passing the `features` object name as `EBS_MALWARE_PROTECTION` and `status` as `ENABLED`.

You can enable GuardDuty-initiated malware scan by running the following AWS CLI command. Make sure to use delegated GuardDuty administrator account's valid *`detector ID`*.

To find the `detectorId` for your account and current Region, see the **Settings** page in the https://console.aws.amazon.com/guardduty/ console, or run the [ListDetectors](#) API.

```
aws guardduty update-detector --detector-id 12abc34d567e8fa901bc2d34e56789f0 /
              --account-ids 555555555555 /
              --features '[{"Name": "EBS_MALWARE_PROTECTION", "Status": "ENABLED"}]'
```

Auto-enable GuardDuty-initiated malware scan for all member accounts

Choose your preferred access method to enable the GuardDuty-initiated malware scan feature for all member accounts. This includes existing member accounts and the new accounts that join the organization.

Console

1. Sign in to the AWS Management Console and open the GuardDuty console at https://console.aws.amazon.com/guardduty/.

   Make sure to use the delegated GuardDuty administrator account credentials.
2. Do one of the following:

   Using the Malware Protection for EC2 page

   1. In the navigation pane, choose **Malware Protection for EC2**.
   2. On the **Malware Protection for EC2** page, choose **Edit** in the **GuardDuty-initiated malware scan** section.
   3. Choose **Enable for all accounts**. This action automatically enables GuardDuty-initiated malware scan for both existing and new accounts in the organization.
   4. Choose **Save**.

   > **Note**
   >
   > It may take up to 24 hours to update the configuration for the member accounts.

   Using the Accounts page

   1. In the navigation pane, choose **Accounts**.
   2. On the **Accounts** page, choose **Auto-enable** preferences before **Add accounts by invitation**.
   3. In the **Manage auto-enable preferences** window, choose **Enable for all accounts** under **GuardDuty-initiated malware scan**.
   4. On the **Malware Protection for EC2** page, choose **Edit** in the **GuardDuty-initiated malware scan** section.
   5. Choose **Enable for all accounts**. This action automatically enables GuardDuty-initiated malware scan for both existing and new accounts in the organization.
   6. Choose **Save**.

> **Note**
>
> It may take up to 24 hours to update the configuration for the member accounts.

Using the Accounts page

1. In the navigation pane, choose **Accounts**.
2. On the **Accounts** page, choose **Auto-enable** preferences before **Add accounts by invitation**.
3. In the **Manage auto-enable preferences** window, choose **Enable for all accounts** under **GuardDuty-initiated malware scan**.
4. Choose **Save**.

If you can't use the **Enable for all accounts** option, see Selectively enable GuardDuty-initiated malware scan for member accounts.

API/CLI

- To selectively enable GuardDuty-initiated malware scan for your member accounts, invoke the updateMemberDetectors API operation using your own *detector ID*.
- The following example shows how you can enable GuardDuty-initiated malware scan for a single member account. To disable a member account, replace `ENABLED` with `DISABLED`.

  To find the `detectorId` for your account and current Region, see the **Settings** page in the https://console.aws.amazon.com/guardduty/ console, or run the ListDetectors API.

  ```
  aws guardduty update-member-detectors --detector-
  id 12abc34d567e8fa901bc2d34e56789f0 --account-ids 111122223333 --features
   '[{"Name": "EBS_MALWARE_PROTECTION", "Status": "ENABLED"}]'
  ```

  You can also pass a list of account IDs separated by a space.
- When the code has successfully executed, it returns an empty list of `UnprocessedAccounts`. If there were any problems changing the detector settings for an account, that account ID is listed along with a summary of the issue.

Enable GuardDuty-initiated malware scan for all existing active member accounts

Choose your preferred access method to enable GuardDuty-initiated malware scan for all the existing active member accounts in the organization.

To configure GuardDuty-initiated malware scan for all existing active member accounts

1. Sign in to the AWS Management Console and open the GuardDuty console at https://console.aws.amazon.com/guardduty/.

   Sign in using the delegated GuardDuty administrator account credentials.
2. In the navigation pane, choose **Malware Protection for EC2**.
3. On the **Malware Protection for EC2**, you can view the current status of the **GuardDuty-initiated malware scan** configuration. Under the **Active member accounts** section, choose **Actions**.
4. From the **Actions** dropdown menu, choose **Enable for all existing active member accounts**.
5. Choose **Save**.

Auto-enable GuardDuty-initiated malware scan for new member accounts

The newly added member accounts must **Enable** GuardDuty before selecting configuring GuardDuty-initiated malware scan. The member accounts managed by invitation can configure GuardDuty-initiated malware scan manually for their accounts. For more information, see [Step 3 - Accept an invitation](#).

Choose your preferred access method to enable GuardDuty-initiated malware scan for new accounts that join your organization.

Console

The delegated GuardDuty administrator account can enable GuardDuty-initiated malware scan for new member accounts in an organization, using either the **Malware Protection for EC2** or **Accounts** page.

To auto-enable GuardDuty-initiated malware scan for new member accounts

1. Open the GuardDuty console at https://console.aws.amazon.com/guardduty/.

   Make sure to use the delegated GuardDuty administrator account credentials.

2. Do one of the following:

   - Using the **Malware Protection for EC2** page:

     1. In the navigation pane, choose **Malware Protection for EC2**.
     2. On the **Malware Protection for EC2** page, choose **Edit** in the **GuardDuty-initiated malware scan**.
     3. Choose **Configure accounts manually**.
     4. Select **Automatically enable for new member accounts**. This step ensures that whenever a new account joins your organization, GuardDuty-initiated malware scan will be automatically enabled for their account. Only the organization delegated GuardDuty administrator account can modify this configuration.
     5. Choose **Save**.

   - Using the **Accounts** page:

     1. In the navigation pane, choose **Accounts**.
     2. On the **Accounts** page, choose **Auto-enable** preferences.
     3. In the **Manage auto-enable preferences** window, select **Enable for new accounts** under **GuardDuty-initiated malware scan**.
     4. Choose **Save**.

API/CLI

- To enable or disable GuardDuty-initiated malware scan for new member accounts, invoke the UpdateOrganizationConfiguration API operation using your own *detector ID*.
- The following example shows how you can enable GuardDuty-initiated malware scan for a single member account. To disable it, see Selectively enable GuardDuty-initiated malware scan for member accounts. If you don't want to enable it for all the new accounts joining the organization, set `AutoEnable` to `NONE`.

  To find the `detectorId` for your account and current Region, see the **Settings** page in the https://console.aws.amazon.com/guardduty/ console, or run the ListDetectors API.

```
aws guardduty update-organization-configuration --detector-
id 12abc34d567e8fa901bc2d34e56789f0 --AutoEnable --features '[{"Name":
 "EBS_MALWARE_PROTECTION", "AutoEnable": NEW}]'
```

You can also pass a list of account IDs separated by a space.

- When the code has successfully executed, it returns an empty list of `UnprocessedAccounts`. If there were any problems changing the detector settings for an account, that account ID is listed along with a summary of the issue.

Selectively enable GuardDuty-initiated malware scan for member accounts

Choose your preferred access method to configure GuardDuty-initiated malware scan for member accounts selectively.

Console

1. Open the GuardDuty console at https://console.aws.amazon.com/guardduty/.
2. In the navigation pane, choose **Accounts**.
3. On the **Accounts** page, review the **GuardDuty-initiated malware scan** column for the status of your member account.
4. Select the account for which you want to configure GuardDuty-initiated malware scan. You can select multiple accounts at a time.
5. From the **Edit protection plans** menu, choose the appropriate option for **GuardDuty-initiated malware scan**.

API/CLI

To selectively enable or disable GuardDuty-initiated malware scan for your member accounts, invoke the updateMemberDetectors API operation using your own *`detector ID`*.

The following example shows how you can enable GuardDuty-initiated malware scan for a single member account.

To find the `detectorId` for your account and current Region, see the **Settings** page in the https://console.aws.amazon.com/guardduty/ console, or run the ListDetectors API.

```
aws guardduty update-member-detectors --detector-id 12abc34d567e8fa901bc2d34e56789f0
 --account-ids 111122223333 --features '[{"Name": "EBS_MALWARE_PROTECTION",
 "Status": "ENABLED"}]'
```

You can also pass a list of account IDs separated by a space.

When the code has successfully executed, it returns an empty list of `UnprocessedAccounts`. If there were any problems changing the detector settings for an account, that account ID is listed along with a summary of the issue.

To selectively enable GuardDuty-initiated malware scan for your member accounts, run the [updateMemberDetectors](https://docs.aws.amazon.com/guardduty/latest/APIReference/API_UpdateMemberDetectors.html) API operation using your own *`detector ID`*. The following example shows how you can enable GuardDuty-initiated malware scan for a single member account.

To find the `detectorId` for your account and current Region, see the **Settings** page in the https://console.aws.amazon.com/guardduty/ console, or run the [ListDetectors](https://docs.aws.amazon.com/guardduty/latest/APIReference/API_ListDetectors.html) API.

```
aws guardduty update-member-detectors --detector-id 12abc34d567e8fa901bc2d34e56789f0
 --account-ids 111122223333 --data-sources '{"MalwareProtection":
{"ScanEc2InstanceWithFindings":{"EbsVolumes":true}}}'
```

You can also pass a list of account IDs separated by a space.

When the code has successfully executed, it returns an empty list of `UnprocessedAccounts`. If there were any problems changing the detector settings for an account, that account ID is listed along with a summary of the issue.

Enable GuardDuty-initiated malware scan for existing accounts in the Organization managed via invitation

The GuardDuty Malware Protection for EC2 service-linked role (SLR) must be created in member accounts. The administrator account can't enable the GuardDuty-initiated malware scan feature in member accounts that are not managed by AWS Organizations.

Presently, you can perform the following steps through the GuardDuty console at https://console.aws.amazon.com/guardduty/ to enable GuardDuty-initiated malware scan for the existing member accounts.

Console

1. Open the GuardDuty console at https://console.aws.amazon.com/guardduty/.

   Sign in using your administrator account credentials.
2. In the navigation pane, choose **Accounts**.
3. Select the member account for which you want to enable GuardDuty-initiated malware scan. You can select multiple accounts at a time.

4. Choose **Actions**.
5. Choose **Disassociate member**.
6. In your member account, choose **Malware Protection** under **Protection plans** on the navigation pane.
7. Choose **Enable GuardDuty-initiated malware scan**. GuardDuty will create an SLR for the member account. For more information on SLR, see Service-linked role permissions for Malware Protection for EC2.
8. In your administrator account account, choose **Accounts** on the navigation pane.
9. Choose the member account that needs to be added back to the organization.
10. Choose **Actions** and then, choose **Add member**.

API/CLI

1. Use administrator account account to run DisassociateMembers API on the member accounts that want to enable GuardDuty-initiated malware scan.
2. Use your member account to invoke UpdateDetector to enable GuardDuty-initiated malware scan.

   To find the `detectorId` for your account and current Region, see the **Settings** page in the https://console.aws.amazon.com/guardduty/ console, or run the ListDetectors API.

   ```
   aws guardduty update-detector --detector-id 12abc34d567e8fa901bc2d34e56789f0
    --data-sources '{"MalwareProtection":{"ScanEc2InstanceWithFindings":
   {"EbsVolumes":true}}}'
   ```
3. Use administrator account account to run the CreateMembers API to add the member back to the organization.

Enabling GuardDuty-initiated malware scan for a standalone account

A standalone account owns the decision to enable or disable a protection plan in their AWS account in a specific AWS Region.

If your account is associated with a GuardDuty administrator account through AWS Organizations, or by the method of invitation, this section doesn't apply to your account. For more information, see Enabling GuardDuty-initiated malware scan in multiple-account environments.

After you enable GuardDuty-initiated malware scan, GuardDuty will initiate a malware scan of the Amazon EBS volume that is attached to the Amazon EC2 instance that was involved in a GuardDuty. For a list of findings that initiate malware scan, see Findings that invoke GuardDuty-initiated malware scan.

Choose your preferred access method to configure GuardDuty-initiated malware scan for a standalone account.

Console

1. Open the GuardDuty console at https://console.aws.amazon.com/guardduty/.
2. In the navigation pane, under **Protection plans**, choose **Malware Protection for EC2**.
3. The Malware Protection for EC2 pane lists the current status of GuardDuty-initiated malware scan for your account. Choose **Enable** to enable GuardDuty-initiated malware scan in this account.
4. Choose **Save** to confirm your selection.

API/CLI

Run the updateDetector API operation using your own regional detector ID and passing the `dataSources` object with `EbsVolumes` set to `true`.

You can also enable GuardDuty-initiated malware scan using AWS CLI by running the following AWS CLI command. Make sure to use your own valid *`detector ID`*.

To find the `detectorId` for your account and current Region, see the **Settings** page in the https://console.aws.amazon.com/guardduty/ console, or run the ListDetectors API.

```
aws guardduty update-detector --detector-id 12abc34d567e8fa901bc2d34e56789f0 --
features [{"Name" : "EBS_MALWARE_PROTECTION", "Status" : "ENABLED"}]'
```

Findings that invoke GuardDuty-initiated malware scan

When GuardDuty detects suspicious behavior that is indicative of malware on an Amazon EC2 instance or a container workload that is running on an Amazon EC2 instance, GuardDuty will generate a finding. If this generated finding belongs to the following list of GuardDuty findings, then GuardDuty will automatically initiate malware scan on the Amazon EBS volumes attached to the Amazon EC2 instance that is involved in the finding. After the scan, if GuardDuty detects malware, then it will also generate one or more Malware Protection for EC2 finding types.

If any of the following GuardDuty findings get generated in your account, GuardDuty will automatically initiate malware scan in the Amazon EBS volume of the potentially compromised Amazon EC2 instance.

- Backdoor:EC2/C&CActivity.B
- Backdoor:EC2/C&CActivity.B!DNS
- Backdoor:EC2/DenialOfService.Dns
- Backdoor:EC2/DenialOfService.Tcp
- Backdoor:EC2/DenialOfService.Udp
- Backdoor:EC2/DenialOfService.UdpOnTcpPorts
- Backdoor:EC2/DenialOfService.UnusualProtocol
- Backdoor:EC2/Spambot
- CryptoCurrency:EC2/BitcoinTool.B
- CryptoCurrency:EC2/BitcoinTool.B!DNS
- DefenseEvasion:Runtime/PtraceAntiDebugging
- DefenseEvasion:Runtime/SuspiciousCommand
- Execution:Runtime/MaliciousFileExecuted
- Execution:Runtime/SuspiciousCommand
- Execution:Runtime/SuspiciousShellCreated
- Execution:Runtime/SuspiciousTool
- Impact:EC2/AbusedDomainRequest.Reputation
- Impact:EC2/BitcoinDomainRequest.Reputation
- Impact:EC2/MaliciousDomainRequest.Reputation
- Impact:EC2/PortSweep
- Impact:EC2/SuspiciousDomainRequest.Reputation
- Impact:EC2/WinRMBruteForce (Outbound only)
- PrivilegeEscalation:Runtime/ElevationToRoot
- Recon:EC2/Portscan
- Trojan:EC2/BlackholeTraffic

- Trojan:EC2/BlackholeTraffic!DNS
- Trojan:EC2/DGADomainRequest.B
- Trojan:EC2/DGADomainRequest.C!DNS
- Trojan:EC2/DNSDataExfiltration
- Trojan:EC2/DriveBySourceTraffic!DNS
- Trojan:EC2/DropPoint
- Trojan:EC2/DropPoint!DNS
- Trojan:EC2/PhishingDomainRequest!DNS
- UnauthorizedAccess:EC2/RDPBruteForce (Outbound only)
- UnauthorizedAccess:EC2/SSHBruteForce (Outbound only)
- UnauthorizedAccess:EC2/TorClient
- UnauthorizedAccess:EC2/TorRelay
- Backdoor:Runtime/C&CActivity.B
- Backdoor:Runtime/C&CActivity.B!DNS
- CryptoCurrency:Runtime/BitcoinTool.B
- CryptoCurrency:Runtime/BitcoinTool.B!DNS
- Execution:Runtime/NewBinaryExecuted
- Execution:Runtime/NewLibraryLoaded
- Execution:Runtime/ReverseShell
- Impact:Runtime/AbusedDomainRequest.Reputation
- Impact:Runtime/BitcoinDomainRequest.Reputation
- Impact:Runtime/CryptoMinerExecuted
- Impact:Runtime/MaliciousDomainRequest.Reputation
- Impact:Runtime/SuspiciousDomainRequest.Reputation
- PrivilegeEscalation:Runtime/CGroupsReleaseAgentModified
- PrivilegeEscalation:Runtime/ContainerMountsHostDirectory
- PrivilegeEscalation:Runtime/DockerSocketAccessed
- PrivilegeEscalation:Runtime/RuncContainerEscape
- PrivilegeEscalation:Runtime/UserfaultfdUsage

- Trojan:Runtime/BlackholeTraffic
- Trojan:Runtime/BlackholeTraffic!DNS
- Trojan:Runtime/DropPoint
- Trojan:Runtime/DropPoint!DNS
- Trojan:Runtime/DGADomainRequest.C!DNS
- Trojan:Runtime/DriveBySourceTraffic!DNS
- Trojan:Runtime/PhishingDomainRequest!DNS
- UnauthorizedAccess:Runtime/MetadataDNSRebind

On-demand malware scan in GuardDuty

On-demand malware scan helps you detect the presence of malware on Amazon Elastic Block Store (Amazon EBS) volumes attached to your Amazon EC2 instances. With no configuration needed, you can start an on-demand malware scan by providing the Amazon Resource Name (ARN) of the Amazon EC2 instance that you want to scan. You can start an on-demand malware scan either through the GuardDuty console or API. Before initiating an on-demand malware scan, you can set your preferred Snapshots retention setting. The following scenarios can help you identify when to use the On-demand malware scan type with GuardDuty:

- You want to detect the presence of malware in your Amazon EC2 instances without enabling GuardDuty-initiated malware scan.
- You have enabled GuardDuty-initiated malware scan and a scan was invoked automatically. After following the recommended remediation for the generated Malware Protection for EC2 finding type, if you want to start a scan on the same resource, you can start an on-demand malware scan after 1 hour has passed from the previous scan start time.

  On-demand malware scan doesn't require that 24 hours have passed from the time the previous malware scan was started. One hour should have passed before initiating an On-demand malware scan on the same resource. To avoid duplicating a malware scan on the same EC2 instance, see Re-scanning previously scanned Amazon EC2 instance.

> **Note**
>
> On-demand malware scan is not included in the 30-day free trial period with GuardDuty. The usage cost applies to the total Amazon EBS volume scanned for each malware scan.

For more information, see Amazon GuardDuty pricing. For information about the cost of creating the Amazon EBS volume snapshots and their retention, see Amazon EBS pricing.

How On-demand malware scan works

With On-demand malware scan, you can start a malware scan request for your Amazon EC2 instance even when it is currently in use. After you start an On-demand malware scan, GuardDuty creates snapshots of the Amazon EBS volumes attached to the Amazon EC2 instance whose Amazon Resource Name (ARN) was provided for the scan. Next, GuardDuty shares these snapshots with the GuardDuty service account. GuardDuty creates encrypted replica EBS volumes from those snapshots in the GuardDuty service account. For more information about how the Amazon EBS volumes are scanned, see How GuardDuty scans EBS volumes for malware detection.

Note

GuardDuty creates the snapshots of the data that has already been written to the Amazon EBS volumes at the point-in-time when you start an On-demand malware scan.

If malware is found and you've enabled the snapshots retention setting, the snapshots of your EBS volume are automatically retained in your AWS account. On-demand malware scan generates the Malware Protection for EC2 finding types. If malware is not found, then regardless of the snapshots retention setting, the snapshots of your EBS volumes are deleted.

GuardDuty uses a global tag key, `GuardDutyExcluded`, that you can add to your Amazon EC2 resources and set the tag value to `true`. This Amazon EC2 resource that has this tag key and value pair will be excluded from the malware scan. Both the scan types (GuardDuty-initiated malware scan and On-demand malware scan) support the global tag. If you start an on-demand malware scan on an Amazon EC2, a scan ID will be generated. However, the scan will be skipped with an `EXCLUDED_BY_SCAN_SETTINGS` reason. For more information, see Reasons for skipping resource during malware scan.

Starting On-demand malware scan in GuardDuty

This section provides a list of prerequisites before initiating an on-demand malware scan and steps to start the scan on a resource for the first time.

As a GuardDuty administrator account, you can start an on-demand malware scan on behalf of your active member accounts that have the following prerequisites set up in their accounts. Standalone accounts and active member accounts in GuardDuty can also start an on-demand malware scan for their own Amazon EC2 instances.

Prerequisites

Before you start an On-demand malware scan, your account must meet the following prerequisites:

- GuardDuty must be enabled in the AWS Regions where you want to start the on-demand malware scan.
- Ensure that the AWS managed policy: AmazonGuardDutyFullAccess_v2 (recommended) is attached to the IAM user or the IAM role. You will need the access key and secret key associated with the IAM user or the IAM role.
- As a delegated GuardDuty administrator account, you have the option to start an on-demand malware scan on behalf of an active member account.
- Before you start an on-demand malware scan, make sure that no scan was started on the same resource in the past 1 hour; otherwise, it will be de-duped. For more information, see Re-scanning previously scanned Amazon EC2 instance.
- If you're a member account that doesn't have the Service-linked role permissions for Malware Protection for EC2, then initiating an on-demand malware scan for an Amazon EC2 instance that belongs to your account, will automatically create the SLR for Malware Protection for EC2.

> **Important**
>
> Ensure that no one deletes the SLR permissions for Malware Protection for EC2 when the malware scan is still in progress. This malware scan could be either started by GuardDuty or started on-demand. Deleting the SLR will prevent the scan from completing successfully, and providing definite scan result.

Start On-demand malware scan

You can start an on-demand malware scan in your account through GuardDuty console or by using AWS CLI. You will need to provide the Amazon EC2 Amazon Resource Name (ARN) for which you want to start the scan. The detailed steps are provided in both console and API/AWS CLI instructions in the following section.

Choose your preferred access method to start an on-demand malware scan.

Console

1. Open the GuardDuty console at https://console.aws.amazon.com/guardduty/.
2. Start the scan using one of the following options:
   a. Using the **Malware Protection for EC2** page:
      i. In the navigation pane, under **Protection plans**, choose **Malware Protection for EC2**.
      ii. On the **Malware Protection for EC2** page, provide the **Amazon EC2 instance ARN**[1] for which you want to start the scan.
   b. Using the **Malware Scans** page:
      i. In the navigation pane, choose **Malware Scans**.
      ii. Choose **Start on-demand scan** and provide the **Amazon EC2 instance ARN**[1] for which you want to start the scan.
      iii. If this is a re-scan, select an **Amazon EC2** instance ID on the **Malware Scans** page.

         Expand the **Start on-demand scan** dropdown and choose **Re-scan selected instance**.
3. After you successfully start a scan using either method, a scan ID gets generated. You can use this scan ID to track the progress of the scan. For more information, see Monitoring malware scan statuses and results.

API/CLI

Invoke StartMalwareScan that accepts the `resourceArn` of the Amazon EC2 instance[1] for which you want to start an on-demand malware scan.

```
aws guardduty start-malware-scan --resource-arn "arn:aws:ec2:us-east-1:555555555555:instance/i-b188560f"
```

After you successfully start a scan, `StartMalwareScan` returns a `scanId`. Invoke DescribeMalwareScans monitor the progress of the started scan.

[1]For information about the format of your Amazon EC2 instance ARN, see Amazon Resource Name (ARN). For Amazon EC2 instances, you can use the following example ARN format by replacing the values for the partition, Region, AWS account ID, and Amazon EC2 instance ID. For information about length of your instance ID, see Resource IDs.

```
arn:aws:ec2:us-east-1:555555555555:instance/i-b188560f
```

AWS Organizations service control policy – Denied access

Using the Service control policies (SCPs) in AWS Organizations, the delegated GuardDuty administrator account can restrict permissions and deny actions such as initiating an on-demand malware scan for Amazon EC2 instance owned by your accounts.

As a GuardDuty member account, when you start an on-demand malware scan for your Amazon EC2 instances, you may receive an error. You can connect with the management account to understand why an SCP was set up for your member account. For more information, see SCP effects on permissions.

Re-scanning previously scanned Amazon EC2 instance

Whether a scan is GuardDuty-initiated or started on-demand, you can start a new on-demand malware scan on the same Amazon EC2 instance after 1 hour from the start time of the previous malware scan. If the new malware scan gets started within 1 hour of initiation of the previous malware scan, your request will result in the following error, and no scan ID will get generated for this request.

```
A scan was started on this resource recently. You can request a scan on the same resource one hour after the previous scan start time.
```

The steps to re-scan the instance remain the same as starting an on-demand malware scan for the first time. For information about the steps, see Start On-demand malware scan.

To track the status of the malware scans, see Monitoring scan statuses and results in Malware Protection for EC2.

Monitoring scan statuses and results in Malware Protection for EC2

After a malware scan is initiated on an Amazon EC2 instance, GuardDuty provides the status and result fields automatically. You can monitor the status through transitions, and view if malware was detected. The following table provides the possible values associated to the malware scan.

Potential values
`Running`, `Completed`, `Skipped`, or `Failed`
`Clean` or `Infected`
`GuardDuty initiated` or `On demand`

*Scan result is populated only when the scan status becomes `Completed`. The scan result `Infected` means that GuardDuty detected the presence of malware.

Scan results for each malware scan has a retention period of 90 days. Choose your preferred access method to track the status of your malware scan.

Console

1. Open the GuardDuty console at https://console.aws.amazon.com/guardduty/.
2. In the navigation pane, choose **EC2 malware scans**.
3. You can filter the malware scans by the following **Properties** available in the *filter search bar*.
   - **Scan ID –** Unique identifier associated with the EC2 malware scan.
   - **Account ID –** AWS account ID where the malware scan initiated.
   - **EC2 instance ARN –** Amazon Resource Name (ARN) associated with the Amazon EC2 instance associated with the scan.

- **Scan status –** The scan status of the EBS volume, such as **Running**, **Skipped**, and **Completed**
- **Scan type –** Indicates whether this was an On-demand malware scan or a GuardDuty-initiated malware scan.

API/CLI

- After the malware scan has a scan result, use DescribeMalwareScans to filter the malware scans on the basis of `EC2_INSTANCE_ARN`, `SCAN_ID`, `ACCOUNT_ID`, `SCAN_TYPE` `GUARDDUTY_FINDING_ID`, `SCAN_STATUS`, and `SCAN_START_TIME`.

  The `GUARDDUTY_FINDING_ID` filter criteria is available when the `SCAN_TYPE` is GuardDuty initiated.

- You can change the example *`filter-criteria`* in the command below. Presently, you can filter on the basis of one `CriterionKey` at a time. The options for `CriterionKey` are `EC2_INSTANCE_ARN`, `SCAN_ID`, `ACCOUNT_ID`, `SCAN_TYPE` `GUARDDUTY_FINDING_ID`, `SCAN_STATUS`, and `SCAN_START_TIME`.

  You can change the *`max-results`* (up to 50) and the *`sort-criteria`*. The `AttributeName` is mandatory and must be `scanStartTime`.

  In the following example, the values in *`red`* are placeholders. Replace them with the values appropriate for your account. For example, replace the example `detector-id` *`60b8777933648562554d637e0e4bb3b2`* with your own valid `detector-id`. If you use the same `CriterionKey` as below, ensure to replace the example `EqualsValue` with your own valid AWS *`scan-id`*.

  ```
  aws guardduty describe-malware-scans --detector-
  id 60b8777933648562554d637e0e4bb3b2 --max-results 1 --sort-criteria
   '{"AttributeName": "scanStartTime", "OrderBy": "DESC"}' --filter-criteria
   '{"FilterCriterion":[{"CriterionKey":"SCAN_ID", "FilterCondition":
  {"EqualsValue":"123456789012"}}] }'
  ```

- The response of this command displays a maximum of one result with details about the affected resource and malware findings (if `Infected`).

GuardDuty service accounts by AWS Region

When a snapshot gets created and shared with a GuardDuty service account, a new event gets created in your CloudTrail logs. This event specifies the corresponding `snapshotId` and `userId` (GuardDuty service account for that AWS Region). For more information, see [How GuardDuty scans EBS volumes for malware detection](#).

The following example is a snippet from a CloudTrail event that shows the request body for the `ModifySnapshotAttribute` request:

```
"requestParameters": {
        "snapshotId": "snap-1234567890abcdef0",
        "createVolumePermission": {
            "add": {
                "items": [
                    {
                        "userId": "111122223333"
                    }
                ]
            }
        },
        "attributeType": "CREATE_VOLUME_PERMISSION"
    }
```

The following table shows the GuardDuty service accounts for each Region. The `userId` is the GuardDuty service account and depends on the selected Region.

AWS Region	Region code	GuardDuty service account ID (userId)
US East (N. Virginia)	us-east-1	652050842985
US East (Ohio)	us-east-2	178123968615
US West (N. California)	us-west-1	669213148797
US West (Oregon)	us-west-2	447226417196
Asia Pacific (Mumbai)	ap-south-1	913179291432

AWS Region	Region code	GuardDuty service account ID (`userId`)
Asia Pacific (Osaka)	ap-northeast-3	089661699081
Asia Pacific (Seoul)	ap-northeast-2	039163547507
Asia Pacific (Tokyo)	ap-northeast-1	874749492622
Asia Pacific (Singapore)	ap-southeast-1	247460962669
Asia Pacific (Sydney)	ap-southeast-2	124839743349
Canada (Central)	ca-central-1	175877067165
Canada West (Calgary)	ca-west-1	894794104037
Europe (Frankfurt)	eu-central-1	002294850712
Europe (Ireland)	eu-west-1	283769539786
Europe (London)	eu-west-2	310125036783
Europe (Paris)	eu-west-3	866607715269
Europe (Stockholm)	eu-north-1	693780578038
China (Beijing)	cn-north-1	448721096076
China (Ningxia)	cn-northwest-1	480864352451
South America (São Paulo)	sa-east-1	546914126324
Asia Pacific (Hyderabad) (Opt-in)	ap-south-2	682251015962
Asia Pacific (Melbourne) (Opt-in)	ap-southeast-4	353488359550

AWS Region	Region code	GuardDuty service account ID (`userId`)
Asia Pacific (Malaysia) (Opt-in)	ap-southeast-5	009160069308
Asia Pacific (Thailand) (Opt-in)	ap-southeast-7	941377115582
Europe (Spain) (Opt-in)	eu-south-2	936182149045
Europe (Zurich) (Opt-in)	eu-central-2	867642063380
Israel (Tel Aviv) (Opt-in)	il-central-1	619233833001
Europe (Milan) (Opt-in)	eu-south-1	977238331021
Asia Pacific (Hong Kong) (Opt-in)	ap-east-1	249472122084
Middle East (Bahrain) (Opt-in)	me-south-1	404001805210
Africa (Cape Town) (Opt-in)	af-south-1	957664736811
Asia Pacific (Jakarta) (Opt-in)	ap-southeast-3	452118225523
Middle East (UAE) (Opt-in)	me-central-1	828603743433
Mexico (Central) (Opt-in)	mx-central-1	557690616787
Asia Pacific (Taipei)	ap-east-2	863518437308
AWS GovCloud (US-East)	us-gov-east-1	226283551151
AWS GovCloud (US-West)	us-gov-west-1	226300430612

Quotas in Malware Protection for EC2

This section includes the quotas associated with using Malware Protection for EC2. For quotas associated with GuardDuty, see GuardDuty quotas.

The following table provides default availability of varied resources when you use Malware Protection for EC2.

Scope	Default	Comments
Extraction and analysis of data in compressed or archived file	5	The maximum number of nested levels allowed in an archived file.
Number of files within an archived file	1000	The maximum number of files that can be scanned within an archive. This count is the sum of the number of files extracted from the archive and the number of files extracted from all the nested archives.
Number of threats	32	The maximum number of threats that you can view in the findings panel. GuardDuty Malware Protection for EC2 may have detected more threat names. If the number of detected threat names is higher than the default value, you can view the JSON details by selecting the **Finding ID** under the finding name in the details panel of the GuardDuty console.

Scope	Default	Comments
Number of files per detected threat	5	The maximum number of files identified per detected threat. For example, if GuardDuty detects 10 files associated with a single threat, the threat will display a maximum of 5 files.
EBS volumes per scan per instance	11	The maximum number of EBS volumes that GuardDuty can scan per EC2 instance. If there are more than 11 EBS volumes that need to be scanned, GuardDuty Malware Protection for EC2 sorts the `deviceName` alphabetically, and selects the first 11 EBS volumes.
EBS volume size	2048 GB	Associated with an Amazon EC2 instance and container workload, GuardDuty Malware Protection for EC2 can scan each Amazon EBS volume that is up to 2048 GB in size. This quota applies to each AWS Region where the support for Malware Protection for EC2 is available.

Scope	Default	Comments
Supported file system types	GuardDuty Malware Protectio n for EC2 can scan the following file system types: • New Technology File System (NTFS) • X File System (XFS) • Second extended (ext2) File System • Fourth extended (ext4) File System • File Allocation Table (FAT) File System • Virtual File Allocation Table (VFAT) File System	N/A.
Scan options tags	50	The maximum number of resource tags that you can add to customize your malware scan options setting. For more information, see Scan options with user-defi ned tags.
Finding retention period	90	The maximum number of days that GuardDuty retains a finding. For the latest information, see Amazon GuardDuty quotas.

Scope	Default	Comments
Malware scan retention period	90	The maximum number of days that GuardDuty Malware Protection for EC2 retains the history of a scan. For more information on viewing recent malware scans, see Monitoring scan statuses and results in Malware Protection for EC2.
Transactions per second (TPS) for On-demand malware scan	1	The number of On-demand malware scan requests that can be initiated per second in each Region.
Burst limit for On-demand malware scan	1	The number of concurrent malware On-demand malware scan requests that can be initiated per second in each Region.

GuardDuty Malware Protection for S3

Malware Protection for S3 helps you detect potential presence of malware by scanning newly uploaded objects to your selected Amazon Simple Storage Service (Amazon S3) bucket. When an S3 object or a new version of an existing S3 object gets uploaded to your selected bucket, GuardDuty automatically starts a malware scan.

Malware Protection for S3 - Overview and Demo

Two approaches to enable Malware Protection for S3

You can enable Malware Protection for S3 when your AWS account enables the GuardDuty service and you use Malware Protection for S3 as a part of the overall GuardDuty experience, or when you want to use the Malware Protection for S3 feature by itself without enabling the GuardDuty service. When you enable Malware Protection for S3 by itself, the GuardDuty documentation refers to it as using Malware Protection for S3 as an independent feature.

Considerations for using Malware Protection for S3 independently

- GuardDuty security findings – Detector ID is a unique identifier that is associated with your account in a Region. When you enable GuardDuty in one or more Regions in an account, a detector ID gets created automatically for this account in each Region where you enable GuardDuty. For more information, see *Detector* in the Concepts and key terms in Amazon GuardDuty document.

  When you enable Malware Protection for S3 independently in an account, that account will **not** have an associated detector ID. This impacts what GuardDuty features may be available to you. For example, when an S3 malware scan detects the presence of malware, no GuardDuty finding will get generated in your AWS account because all GuardDuty findings are associated with a detector ID.

- Checking if the scanned object is malicious – By default, GuardDuty publishes the malware scan results to your default Amazon EventBridge event bus and an Amazon CloudWatch namespace. When you enable tagging at the time of enabling Malware Protection for S3 for a bucket, the scanned S3 object gets a tag that mentions the scan result. For more information about tagging, see Optional tagging of objects based on scan result.

General considerations for enabling Malware Protection for S3

The following general consideration apply whether you use Malware Protection for S3 independently or as a part of the GuardDuty experience:

- You can enable Malware Protection for S3 for an Amazon S3 bucket that belongs to your own account. As a delegated GuardDuty administrator account you can't enable this feature in an Amazon S3 bucket that belongs to a member account.
- You can enable this feature in the S3 buckets that belong to the same Region that is currently selected in the GuardDuty console. GuardDuty doesn't support enabling this feature in cross-Region S3 buckets.
- As a delegated GuardDuty administrator account, you will receive an Amazon EventBridge notification each time there is a change in the Viewing and understanding protected bucket status of an S3 bucket that one of your organization's member account configured for this feature.

Contents

- Pricing and usage cost for Malware Protection for S3
- How does Malware Protection for S3 work?
- Capabilities of Malware Protection for S3
- (Optional) Get started with GuardDuty Malware Protection for S3 independently (console only)
- Configuring Malware Protection for S3 for your bucket
- Steps after enabling Malware Protection for S3
- Using tag-based access control (TBAC) with Malware Protection for S3
- Viewing and understanding protected bucket status
- Troubleshooting Malware Protection plan status
- Monitoring S3 object scans in Malware Protection for S3
- Editing Malware Protection plan for a protected bucket
- Disabling Malware Protection for S3 for a protected bucket
- Supportability of Amazon S3 features
- Quotas in Malware Protection for S3

Pricing and usage cost for Malware Protection for S3

The pricing in Malware Protection for S3 works differently than other protection plans in GuardDuty. While most of the GuardDuty protection plans follow a 30-day short term free trial, Malware Protection for S3 follows 12 months Free Tier plan in AWS. For information about GuardDuty pricing, see Pricing in GuardDuty.

The following list provides the pricing costs associated with using Malware Protection for S3.

Free Tier plan (scanning cost)

Each AWS account gets a 12-month Free Tier that includes usage up to a specific limit per month for each Region. If your usage goes beyond the specified limit, you will start incurring the usage cost for the exceeded limit. For information about the specified limits and a pricing example, see GuardDuty protection plans pricing.

- All existing AWS accounts are eligible to use the 12-month Free Tier for this feature that starts from June 11, 2024 and ends on June 11, 2025. This extended 12-month Free Tier for your account applies to using Malware Protection for S3, and no other AWS service or another GuardDuty feature.

  If an existing AWS account starts using Malware Protection for S3 after June 11, 2025 or after the 12-month Free Tier of the account ends, then you will start incurring the associated usage cost.
- If you have a new AWS account and your 12-month Free Tier starts after the general availability (June 11, 2024) of Malware Protection for S3, then your 12-month Free Tier period for this feature will be the same as the 12-month Free Tier period for your account.

For information about the usage cost after enabling Malware Protection for S3, see Reviewing usage cost for Malware Protection for S3.

S3 Object Tagging usage cost

When you enable Malware Protection for S3, it is optional to enable tagging for your scanned S3 objects. When you choose to enable S3 Object Tagging, there is an associated usage cost. For more information about the costs, see Management & insights tab on the *Amazon S3 pricing page*.

S3 Object Tagging usage cost is **not included** in the Free Tier plan.

Amazon S3 APIs - GET and PUT usage cost

You will incur usage cost when GuardDuty runs the Amazon S3 APIs based on the IAM role. For example, after assuming the IAM role, GuardDuty runs the `PutObject` API to add the test object to your selected bucket. This helps GuardDuty assess the enabled status of the feature.

For information about pricing of S3 API calls in your AWS Region, see [Requests & data retrievals under the Storage & requests tab] on the *Amazon S3 pricing page*.

Reviewing usage cost for Malware Protection for S3

Your account starts incurring usage cost when you use Malware Protection for S3 beyond the specific limit under the Free Tier plan, or when your account's 12-month Free Tier plan ends. For information about the Free Tier plan, see Pricing and usage cost for Malware Protection for S3.

The GuardDuty console doesn't support reviewing the Malware Protection for S3 usage cost. To view the usage cost, navigate to **Cost Explorer** in the https://console.aws.amazon.com/costmanagement/ console. For information about AWS account billing, see the AWS Billing User Guide.

For information about estimated usage cost in GuardDuty, see Estimating usage cost.

How does Malware Protection for S3 work?

This section describes components of Malware Protection for S3, how it works after you enable it for an S3 bucket, and how you can review the malware scan status and result.

Overview

You can enable Malware Protection for S3 for an Amazon S3 bucket that belongs to your own AWS account. GuardDuty provides you flexibility to enable this feature for your entire bucket, or limit the scope of the malware scan to specific object prefixes where GuardDuty scans each uploaded object that starts with one of the selected prefixes. You can add up to 5 prefixes. When you enable the feature for an S3 bucket, then that bucket is called a **protected bucket**.

IAM role permissions

Malware Protection for S3 uses an IAM role that permits GuardDuty to perform the malware scan actions on your behalf. These actions include being notified of the newly uploaded objects in your selected bucket, scanning those objects, and optionally adding tags to your scanned objects. This is a prerequisite to configuring your S3 bucket with this feature.

You have the option to either update an existing IAM role, or create a new role for this purpose. When you enable Malware Protection for S3 for more than one bucket, you can update the existing IAM role to include the other bucket name, as needed. For more information, see Create or update IAM role policy.

Optional tagging of objects based on scan result

At the time of enabling Malware Protection for S3 for your bucket, there is an optional step to enable tagging for scanned S3 objects. The IAM role already includes the permission to add tags to your object after the scan. However, GuardDuty will add tags only when you enable this option at the time of setup.

You must enable this option before an object gets uploaded. After the scan ends, GuardDuty adds a predefined tag to the scanned S3 object with the following key:value pair:

`GuardDutyMalwareScanStatus:`*`Potential scan result`*

The potential scan result tag values include `NO_THREATS_FOUND`, `THREATS_FOUND`, `UNSUPPORTED`, `ACCESS_DENIED`, and `FAILED`. For more information about these values, see the section called "S3 object potential scan status and result status".

Enabling tagging is one of the ways to know about the S3 object scan result. You can further use these tags to add a tag-based access control (TBAC) S3 resource policy so that you can take actions on the potentially malicious objects. For more information, see Adding TBAC on S3 bucket resource.

We recommend you to enable tagging at the time of configuring Malware Protection for S3 for your bucket. If you enable tagging after an object gets uploaded and potentially the scan initiates, GuardDuty will not be able to add tags to the scanned object. For information about associated S3 Object Tagging cost, see Pricing and usage cost for Malware Protection for S3.

Process after you enable Malware Protection for S3 for a bucket

After you enable Malware Protection for S3, a **Malware Protection plan resource** gets created exclusively for the selected S3 bucket. This resource is associated with a Malware Protection plan ID, a unique identifier for your protected resource. By using one of the IAM permissions, GuardDuty then creates and manages an EventBridge managed rule by the name of `DO-NOT-DELETE-AmazonGuardDutyMalwareProtectionS3*`.

How GuardDuty handles your data - guardrails for data protection

Malware Protection for S3 listens to the Amazon EventBridge notifications. When an object gets uploaded to the selected bucket or one of the prefixes, GuardDuty downloads that object from S3 bucket by using an AWS PrivateLink and then reads, decrypts, and scans it in an isolated environment in the same Region. The scanning environment runs in a locked down virtual private cloud (VPC) with no internet access. The VPC is attached to a DNS Firewall rule group that allows communication only to the allowslisted domains that AWS owns. For the duration of the scan, GuardDuty temporarily stores the downloaded S3 object within the scanning environment that is encrypted with AWS Key Management Service (AWS KMS) keys.

> **Note**
>
> By default, all the Amazon S3 APIs listed under the Object Created Event type in the *Amazon S3 User Guide*, will initiate the Malware Protection for S3 scan. These *Event types* include PutObject, POST Object, CopyObject, and CompleteMultipartUpload.

For information about GuardDuty malware detection methodology and the scan engines that it uses, see GuardDuty malware detection scan engine.

After the malware scan completes, GuardDuty processes the scan metadata with the scan status and then deletes the downloaded copy of the object.

GuardDuty cleans the scanning environment each time before a new scan begins. GuardDuty uses contingent authorization for operator access to the scanning environment, and every access request is reviewed, approved, and audited.

Reviewing S3 object scan status and result

GuardDuty publishes the S3 object scan result event to Amazon EventBridge default event bus. GuardDuty also sends the scan metrics such as number of objects scanned and bytes scanned to Amazon CloudWatch. If you enabled tagging, then GuardDuty will add the predefined tag `GuardDutyMalwareScanStatus` and a potential scan result as the tag value.

> **Important**
>
> GuardDuty uses at-least-once delivery, which means you might receive multiple scan results for the same object. We recommend designing your applications to handle duplicate results. You're billed only once for each scanned object.

For more information, see Monitoring S3 object scans in Malware Protection for S3.

Reviewing generated findings

Reviewing the findings depends on whether or not you are using Malware Protection for S3 with GuardDuty. Consider the following scenarios:

Using Malware Protection for S3 when you have GuardDuty service enabled (detector ID)

If the malware scan detects a potentially malicious file in an S3 object, GuardDuty will generate an associated finding. You can view the finding details and use the recommended steps to potentially remediate the finding. Based on your Export findings frequency, the generated finding gets exported to an S3 bucket and EventBridge event bus.

For information about the finding type that would get generated, see Malware Protection for S3 finding type.

Using Malware Protection for S3 as an independent feature (no detector ID)

GuardDuty will not be able to generate findings because there is no associated detector ID. To know the S3 object malware scan status, you can view the scan result that GuardDuty automatically publishes to your default event bus. You can also view the CloudWatch metrics to assess the number of objects and bytes that GuardDuty attempted to scan. You can set up CloudWatch alarms to get notified about the scan results. If you have enabled S3 Object Tagging, you can also view the malware scan status by checking the S3 object for the `GuardDutyMalwareScanStatus` tag key and the scan result tag value.

For information about the S3 object scan status and result, see Monitoring S3 object scans in Malware Protection for S3.

Capabilities of Malware Protection for S3

The following list provides an overview of what you can expect or do after enabling Malware Protection for S3 for your bucket:

- **Choose what to scan** – Scan files as they get uploaded to all or specific prefixes (up to 5) associated with your selected S3 bucket.
- **Automatic scans on uploaded objects** – Once you enable Malware Protection for S3 for a bucket, GuardDuty will automatically start a scan to detect potential malware in a newly uploaded object.
- **Enable through console, by using API/AWS CLI, or AWS CloudFormation** – Choose a preferred method to enable Malware Protection for S3.

  You can enable Malware Protection for S3 by using Infrastructure as code (IaC) platforms such as *Terraform*. For more information, see Resource: `aws_guardduty_malware_protection_plan`.
- **Supported file formats, Malware Protection for S3 quotas, and Amazon S3 features** – Malware Protection for S3 supports all file formats that you can upload to the S3 buckets. If the uploaded file is password-protected, then GuardDuty will skip scanning the file. For information about the quotas related to object size, maximum archive depth level, and other details, see Quotas in Malware Protection for S3.

  For information about whether or not an Amazon S3 feature is supported, see Supportability of Amazon S3 features.
- **Supports tagging scanned S3 object** – When you enable Optional tagging of objects based on scan result, then after each malware scan, GuardDuty will add a tag that indicates the scan status. You can use this tag to set up tag-based access control (TBAC) for the S3 objects. For example, you can restrict access to the S3 objects that are indicated as malicious and have the tag value as `THREATS_FOUND`.
- **Amazon EventBridge notifications** – GuardDuty sends events to Amazon EventBridge when the Malware Protection plan resource status changes, or a malware scan of the S3 object completes. These events are sent to the default event bus. You can use EventBridge and these events to write rules that take actions, such as monitoring when these events happen. For more information, see Monitoring S3 object scans with Amazon EventBridge.
- **CloudWatch metrics** – View CloudWatch metrics to enable alarms on certain malware scan status. For more information, see S3 object scan status metrics in CloudWatch.

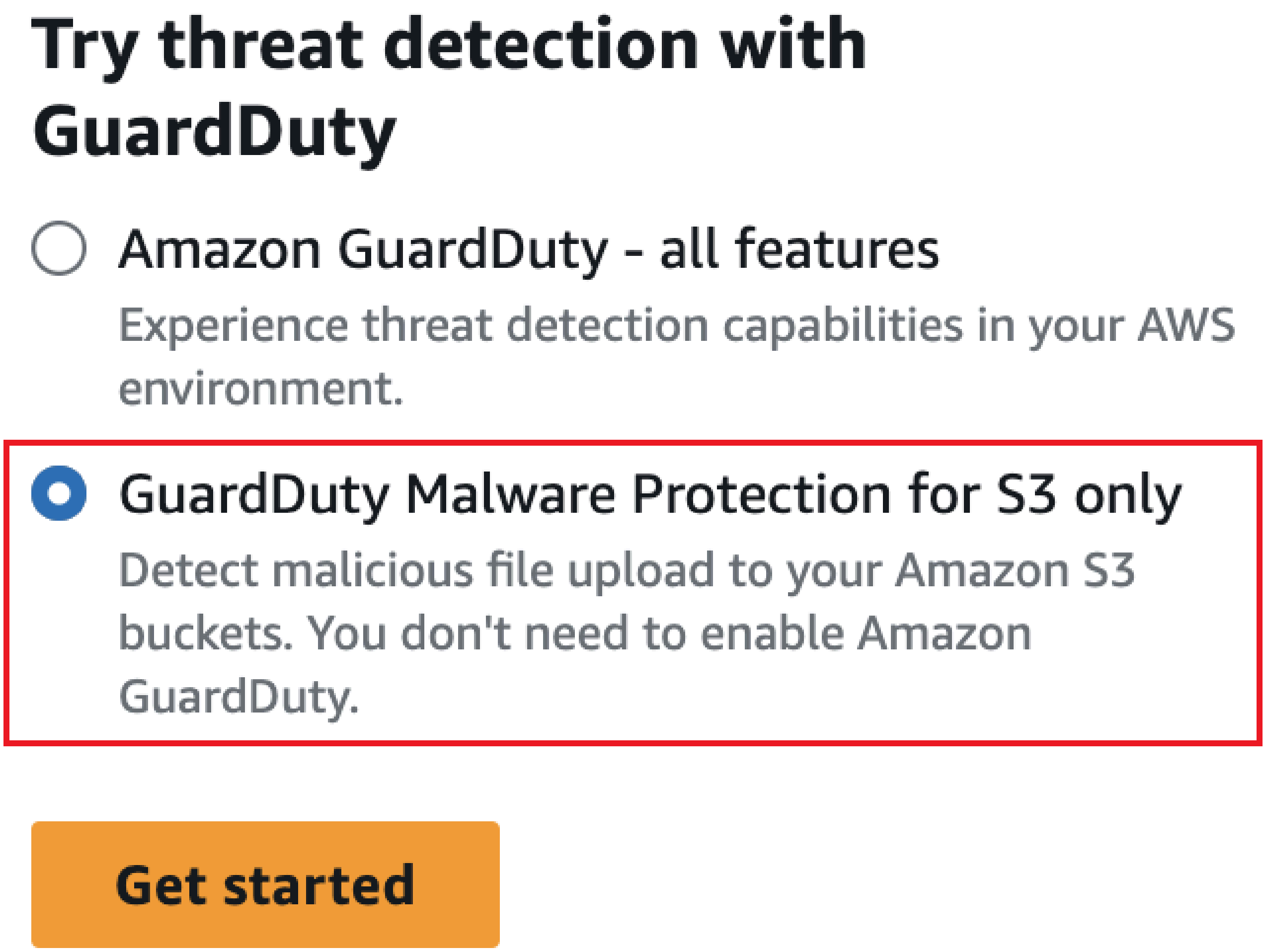
(Optional) Get started with GuardDuty Malware Protection for S3 independently (console only)

Use this optional step when you want to get started with Malware Protection for S3 threat detection option independent of the GuardDuty status in your AWS account.

If you also want to use other dedicated protection plans in GuardDuty, you must get started with the Amazon GuardDuty service. For information about GuardDuty protection plans, see Features of GuardDuty. When you have already enabled GuardDuty in your account, then you can skip this step and continue with Configuring Malware Protection for S3 for your bucket.

Steps to get started with Malware Protection for S3 only threat detection

1. Sign in to the AWS Management Console and open the GuardDuty console at https://console.aws.amazon.com/guardduty/.
2. Select **GuardDuty Malware Protection for S3 only**. This helps you detect if a newly uploaded file in your Amazon Simple Storage Service (Amazon S3) bucket potentially contains malware.



3. Choose **Get started**. You can now continue with steps under [Configuring Malware Protection for S3 for your bucket](#).

Configuring Malware Protection for S3 for your bucket

For Malware Protection for S3 to scan and (optionally) add tags to your S3 objects, you can use service roles that has the necessary permissions to perform malware scan actions on your behalf. For more information about using service roles to enable malware protection for S3, see [Service Access](#). This role is different from the [GuardDuty Malware Protection service-linked role](#).

If you prefer to use IAM roles, you can attach an IAM role that includes the required permissions to scan and (optionally) add tags to your S3 objects. GuardDuty then assumes this IAM role to perform these actions on your behalf. You will need this IAM role name at the time of enabling this protection plan for your Amazon S3 bucket.

If you are using IAM roles, for each time you want to protect an Amazon S3 bucket, you must perform both the steps listed in this section.

To enable Malware Protection for S3, you will need details such as S3 bucket name, object prefixes if you want to focus the protection for specific prefixes, and the IAM role name with required permissions.

The steps remain the same whether you get started with Malware Protection for S3 independently or enable it as a part of the GuardDuty service.

Topics

1. Create or update IAM role policy
2. Enabling Malware Protection for S3 for your bucket
3. Troubleshooting IAM role permissions error

Enabling Malware Protection for S3 for your bucket

This section provides detailed steps on how to enable Malware Protection for S3 for a bucket in your own account. Before you proceed, review the following considerations:

- When you enable this protection plan using the GuardDuty console, it includes the step to create a new role or use an existing role under the **Service access** section.
- When you enable this protection plan using the GuardDuty API or CLI, then you must Create or update IAM role policy before proceeding further.
- Regardless of how you enable this protection plan, you must have the required Permissions to create a Malware Protection plan resource.

Considering Amazon S3 bucket throttling

S3 Throttling might limit the rate at which data can be transferred to or from your Amazon S3 buckets. This can potentially delay malware scans of your newly uploaded objects.

If you expect high volumes of GET and PUT requests to your S3 buckets, consider implementing measures to prevent throttling. For information on how to do this, see Prevent Amazon S3 throttling in the *Amazon Athena User Guide*.

Permissions to create a Malware Protection plan resource

When you enable Malware Protection for S3 for an Amazon S3 bucket, GuardDuty creates a Malware Protection plan resource that acts as an identifier for the bucket's protection plan. If you are not already using the AWS managed policy: AmazonGuardDutyFullAccess_v2 (recommended), then you must add the following permissions to create this resource:

- `guardDuty:CreateMalwareProtectionPlan`
- `iam:PassRole`

You can use the following custom policy example and replace the *`placeholder values`* with the values appropriate for your account:

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "iam:PassRole"
            ],
            "Resource": "arn:aws:iam::111122223333:role/role-name",
            "Condition": {
                "StringEquals": {
                    "iam:PassedToService": "malware-protection-
plan.guardduty.amazonaws.com"
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": [
                "guardduty:CreateMalwareProtectionPlan"
            ],
            "Resource": "*"
        }
    ]
}
```

Enabling Malware Protection for S3 by using GuardDuty console

The following sections provide a step-by-step walkthrough as you will experience in the GuardDuty console.

To enable Malware Protection for S3 by using GuardDuty console

Enter S3 bucket details

Use the following steps to provide the Amazon S3 bucket details:

1. Sign in to the AWS Management Console and open the GuardDuty console at https://console.aws.amazon.com/guardduty/.
2. By using the AWS Region selector in the upper-right corner of the page, select the Region where you want to enable Malware Protection for S3.
3. In the navigation pane, choose **Malware Protection for S3**.
4. In the **Protected buckets** section, choose **Enable** to enable Malware Protection for S3 for an S3 bucket that belongs to your own AWS account.
5. Under **Enter S3 bucket details**, enter the **Amazon S3 bucket** name. Alternatively, choose **Browse S3** to select an S3 bucket.

   The AWS Region of the S3 bucket and the AWS account where you enable Malware Protection for S3 must be the same. For example, if your account belongs to the `us-east-1` Region, then your Amazon S3 bucket Region must also be `us-east-1`.
6. Under **Prefix**, you can select either **All the objects in the S3 bucket** or **Objects beginning with a specific prefix**.
   - Select **All the objects in the S3 bucket** when you want GuardDuty can scan all the newly uploaded objects in the selected bucket.
   - Select **Objects beginning with a specific prefix** when you want scan the newly uploaded objects that belong to a specific prefix. This option helps you focus the scope of the malware scan on the selected object prefixes only. For more information about using prefixes, see Organizing objects in Amazon S3 console by using folders in the *Amazon S3 User Guide*.

     Choose **Add prefix** and enter prefix. You can add up to five prefixes.

Enable tagging for scanned objects

This is an **optional** step. When you enable the tagging option before an object gets uploaded to your bucket, then after completing the scan, GuardDuty will add a predefined tag with key as `GuardDutyMalwareScanStatus` and the value as the scan result. To use Malware Protection for S3 optimally, we recommend to enable the option to add tag to the S3 objects after the scan ends. Standard S3 Object Tagging cost applies. For more information, see Pricing and usage cost for Malware Protection for S3.

Why should you enable tagging?

- Enabling tagging is one of the ways to know about the malware scan result. For information about an S3 malware scan result, see Monitoring S3 object scans in Malware Protection for S3.
- Set up tag-based access control (TBAC) policy on your S3 bucket that contains the potentially malicious object. For information about considerations and how to implement tag-based access control (TBAC), see Using tag-based access control (TBAC) with Malware Protection for S3.

Considerations for GuardDuty to add a tag to your S3 object:

- By default, you can associate up to 10 tags with an object. For more information, see Categorizing your storage using tags in the *Amazon S3 User Guide*.

  If all 10 tags are already in use, GuardDuty can't add the predefined tag to the scanned object. GuardDuty also publishes the scan result to your default EventBridge event bus. For more information, see Monitoring S3 object scans with Amazon EventBridge.
- When the selected IAM role doesn't include the permission for GuardDuty to tag the S3 object, then even with tagging enabled for your protected bucket, GuardDuty will be unable to add tag to this scanned S3 object. For more information about the required IAM role permission for tagging, see Create or update IAM role policy.

  GuardDuty also publishes the scan result to your default EventBridge event bus. For more information, see Monitoring S3 object scans with Amazon EventBridge.

To select an option under Tag scanned objects

- When you **want** GuardDuty to add tags to your scanned S3 objects, select **Tag objects**.

- When you **don't want** GuardDuty to add tags to your scanned S3 objects, select **Do not tag objects**.

Service access

Use the following steps to choose an existing service role or create a new service role that has the necessary permissions to perform malware scan actions on your behalf. These actions may include scanning the newly uploaded S3 objects and (optionally) adding tags to those objects. For information about the permissions that this role will have, see Create or update IAM role policy.

In the **Service access** section, you can do one of the following:

1. **Create and use a new service role** — You can use create a new service role that has the necessary permissions to perform malware scan.

   Under the **Role name** you can choose to use the name pre-populated by GuardDuty or enter a meaningful name of your choice to identify the role. For example `GuardDutyS3MalwareScanRole`. The Role name must be 1-64 characters. Valid characters are a-z, A-Z, 0-9, and '+=,.@-_' characters.
2. **Use an existing service role** — You can choose an existing service role from the **Service role name** list.

   a. Under **Policy template** you can view the policy for your S3 bucket. Make sure that you entered or selected an S3 bucket in the **Enter S3 bucket** details section.

   b. Under **Service role name** choose a service role from the list of service roles.

You can make changes to the policy based on your requirements For more details on how you can create or update an IAM role, see Create or update IAM role policy.

For issues with IAM role permissions, see Troubleshooting IAM role permissions error.

(Optional) Tag Malware Protection plan ID

This is an optional step that helps you add tags to the Malware Protection plan resource that would get created for your S3 bucket resource.

Each tag has two parts: A tag key and an optional tag value. For more information about tagging and its benefits, see Tagging AWS resources.

To add tags to your Malware Protection plan resource

1. Enter **Key** and an optional **Value** for the tag. Both tag key and tag value are case sensitive. For information about names of tag key and tag value, see Tag naming limits and requirements.
2. To add more tags to your Malware Protection plan resource, choose **Add new tag** and repeat the previous step. You can add up to 50 tags to each resource.
3. Choose **Enable**.

Enabling Malware Protection for S3 by using API/CLI

This section includes the steps for when you want to enable Malware Protection for S3 programmatically in your AWS environment. This requires the IAM role Amazon Resource Name (ARN) that you created in this step - Create or update IAM role policy.

To enable Malware Protection for S3 programmatically by using API/CLI

- **By using the API**

  Run the CreateMalwareProtectionPlan to enable Malware Protection for S3 for a bucket that belongs to your own account.

- **By using AWS CLI**

  Depending on how you want to enable Malware Protection for S3, the following list provides AWS CLI example commands for specific use case. When you run these commands, replace the *placeholder examples shown in red*, with the values that are appropriate for your account.

  AWS CLI example commands

  - Use the following AWS CLI command to enable Malware Protection for S3 for a bucket with no tagging for scanned S3 objects:

    ```
    aws guardduty create-malware-protection-plan --role
     "arn:aws:iam::111122223333:role/role-name" --protected-resource
     "S3Bucket"={"BucketName"="amzn-s3-demo-bucket1"}
    ```

  - Use the following AWS CLI command to enable Malware Protection for S3 for a bucket with specific object prefixes and no tagging for scanned S3 objects:

```
aws guardduty create-malware-protection-plan --role
 "arn:aws:iam::111122223333:role/role-name" --protected-resource '{"S3Bucket":
{"BucketName":"amzn-s3-demo-bucket1", "ObjectPrefixes": ["Object1","Object1"]}}'
```

- Use the following AWS CLI command to enable Malware Protection for S3 for a bucket with scanned S3 object tagging enabled:

```
aws guardduty create-malware-protection-plan --role
 "arn:aws:iam::111122223333:role/role-name" --protected-resource
 "S3Bucket"={"BucketName"="amzn-s3-demo-bucket1"} --actions
 "Tagging"={"Status"="ENABLED"}
```

After you run these commands successfully, a unique Malware Protection plan ID will get generated. To perform actions such as updating or disabling the protection plan for your bucket, you will need this Malware Protection plan ID.

For issues with IAM role permissions, see Troubleshooting IAM role permissions error.

Create or update IAM role policy

For Malware Protection for S3 to scan and (optionally) add tags to your S3 objects, you can use service roles that has the necessary permissions to perform malware scan actions on your behalf. For more information about using service roles to enable malware protection for S3, see Service Access. This role is different from the GuardDuty Malware Protection service-linked role.

If you prefer to use IAM roles, you can attach an IAM role that includes the required permissions to scan and (optionally) add tags to your S3 objects. You must create an IAM role or update an existing role to include these permissions. Because these permissions are required for each Amazon S3 bucket for which you enable Malware Protection for S3, you need to perform this step for each Amazon S3 bucket that you to protect.

The following list explains how certain permissions help GuardDuty perform the malware scan on your behalf:

- Allow Amazon EventBridge actions to create and manage the EventBridge managed rule so that Malware Protection for S3 can listen to your S3 object notifications.

  For more information, see Amazon EventBridge managed rules in the *Amazon EventBridge User Guide*.

- Allow Amazon S3 and EventBridge actions to send notification to EventBridge for all events in this bucket

  For more information, see Enabling Amazon EventBridge in the *Amazon S3 User Guide*.
- Allow Amazon S3 actions to access the uploaded S3 object and add a predefined tag, `GuardDutyMalwareScanStatus`, to the scanned S3 object. When using an object prefix, add an `s3:prefix` condition on the targeted prefixes only. This prevents GuardDuty from accessing all the S3 objects in your bucket.
- Allow KMS key actions to access the object before scanning and putting a test object on buckets with the supported DSSE-KMS and SSE-KMS encryption.

> **Note**
>
> This step is required each time you enable Malware Protection for S3 for a bucket in your account. If you already have an existing IAM role, you can update its policy to include the details of another Amazon S3 bucket resource. The Adding IAM policy permissions topic provides an example on how to do this.

Use the following policies to create or update an IAM role.

Policies

- Adding IAM policy permissions
- Adding Trust relationship policy

Adding IAM policy permissions

You can choose to update the inline policy of an existing IAM role, or create a new IAM role. For information about the steps, see Creating an IAM role or Modifying a role permissions policy in the *IAM User Guide*.

Add the following permissions template to your preferred IAM role. Replace the following placeholder values with appropriate values associated with your account:

- For *`amzn-s3-demo-bucket`*, replace with your Amazon S3 bucket name.

To use the same IAM role for more than one S3 bucket resource, update an existing policy as displayed in the following example:

```
                    ...
                    ...
                    "Resource": [
                        "arn:aws:s3:::amzn-s3-demo-bucket/*",
                        "arn:aws:s3:::amzn-s3-demo-bucket2/*"
                    ],
                    ...
                    ...
```

Make sure to add a comma (,) before adding a new ARN associated with the S3 bucket. Do this wherever you refer to an S3 bucket `Resource` in the policy template.

- For *111122223333*, replace with your AWS account ID.
- For *us-east-1*, replace with your AWS Region.
- For *APKAEIBAERJR2EXAMPLE*, replace with your customer managed key ID. If your S3 bucket is encrypted by using an AWS KMS key, we add the relevant permissions if you choose the Create a new role option when configuring malware protection for your bucket.

```
"Resource": "arn:aws:kms:us-east-1:111122223333:key/*"
```

IAM role policy template

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [{
            "Sid": "AllowManagedRuleToSendS3EventsToGuardDuty",
            "Effect": "Allow",
            "Action": [
                "events:PutRule",
                "events:DeleteRule",
                "events:PutTargets",
                "events:RemoveTargets"
```

```
            ],
            "Resource": [
                "arn:aws:events:us-east-1:111122223333:rule/DO-NOT-DELETE-
AmazonGuardDutyMalwareProtectionS3*"
            ],
            "Condition": {
                "StringLike": {
                    "events:ManagedBy": "malware-protection-
plan.guardduty.amazonaws.com"
                }
            }
        },
        {
            "Sid": "AllowGuardDutyToMonitorEventBridgeManagedRule",
            "Effect": "Allow",
            "Action": [
                "events:DescribeRule",
                "events:ListTargetsByRule"
            ],
            "Resource": [
                "arn:aws:events:us-east-1:111122223333:rule/DO-NOT-DELETE-
AmazonGuardDutyMalwareProtectionS3*"
            ]
        },
        {
            "Sid": "AllowPostScanTag",
            "Effect": "Allow",
            "Action": [
                "s3:PutObjectTagging",
                "s3:GetObjectTagging",
                "s3:PutObjectVersionTagging",
                "s3:GetObjectVersionTagging"
            ],
            "Resource": [
                "arn:aws:s3:::amzn-s3-demo-bucket/*"
            ]
        },
        {
            "Sid": "AllowEnableS3EventBridgeEvents",
            "Effect": "Allow",
            "Action": [
                "s3:PutBucketNotification",
                "s3:GetBucketNotification"
            ],
```

```
            "Resource": [
                "arn:aws:s3:::amzn-s3-demo-bucket"
            ]
        },
        {
            "Sid": "AllowPutValidationObject",
            "Effect": "Allow",
            "Action": [
                "s3:PutObject"
            ],
            "Resource": [
                "arn:aws:s3:::amzn-s3-demo-bucket/malware-protection-resource-
validation-object"
            ]
        },
        {
            "Sid": "AllowCheckBucketOwnership",
            "Effect": "Allow",
            "Action": [
                "s3:ListBucket"
            ],
            "Resource": [
                "arn:aws:s3:::amzn-s3-demo-bucket"
            ]
        },
        {
           "Sid": "AllowMalwareScan",
            "Effect": "Allow",
            "Action": [
                "s3:GetObject",
                "s3:GetObjectVersion"
            ],
            "Resource": [
                "arn:aws:s3:::amzn-s3-demo-bucket/*"
            ]
        },
        {
            "Sid": "AllowDecryptForMalwareScan",
            "Effect": "Allow",
            "Action": [
                "kms:GenerateDataKey",
                "kms:Decrypt"
            ],
```

```
            "Resource": "arn:aws:kms:us-
east-1:111122223333:key/APKAEIBAERJR2EXAMPLE",
            "Condition": {
                "StringLike": {
                    "kms:ViaService": "s3.us-east-1.amazonaws.com"
                }
            }
        }
    ]
}
```

Adding Trust relationship policy

Attach the following trust policy to your IAM role. For information about steps, see Modifying a role trust policy.

JSON

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "malware-protection-plan.guardduty.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
```

Troubleshooting IAM role permissions error

When enabling Malware Protection for S3, GuardDuty checks if your IAM service role has the necessary permissions to validate Amazon S3 bucket ownership. If these permissions are missing or incorrectly configured, you might get the following message:

```
"message": "The request was rejected because provided IAM role does not have the
 required permissions to validate S3 bucket ownership."
```

```
"type": "InvalidInputException"
```

The following scenarios can help you troubleshoot this error:

Missing IAM role permissions

- The IAM role must have the required permissions to allow Malware Protection for S3 to assume the role.
- GuardDuty validates the bucket ownership with the `"s3:ListBucket"` permission. This must be present in the IAM role that you use.

For information about the permissions, see Create or update IAM role policy.

IAM role availability

- When you create a new IAM role, allow a few minutes for the changes to reach eventual consistency before enabling Malware Protection for S3. If you attempt to enable the protection plan immediately after creating the role, the validation might fail.
- For Infrastructure as Code (IaC) deployments, GuardDuty recommends declaring a resource dependency to ensure the IAM role reaches eventual consistency.

  For sample templates on how to do this, see GuardDuty GitHub repository.

Cross-region enablement

Ensure your Amazon S3 bucket is in the same Region where you are enabling Malware Protection for S3 in GuardDuty.

Steps after enabling Malware Protection for S3

This section lists the steps that you may take after enabling Malware Protection for S3 for a bucket. The following steps are listed in an order that will help you navigate through the next steps:

To follow after you enable Malware Protection for S3 for your bucket

1. **Add tag-based access control (TBAC) resource policy –** When you enable tagging, then before an object gets uploaded to your selected bucket, ensure to add the TBAC policy to your S3 bucket resource. For more information, see Adding TBAC on S3 bucket resource.
2. **Monitor Malware Protection plan status –** Monitor the **Status** column for each protected bucket. For information about potential statuses and what they mean, see Viewing and understanding protected bucket status.

3. **Upload an object**:

   1. Open the Amazon S3 console at https://console.aws.amazon.com/s3/.
   2. Upload a file to the S3 bucket or the object prefix for which you enabled this feature. For steps to upload a file, see Upload an object to your bucket in the *Amazon S3 User Guide*.

4. **Monitor S3 object scan status and scan result** – This step includes information about how to check the malware scan status of the S3 object.

Enabled both GuardDuty and Malware Protection for S3	Enabled Malware Protection for S3 only
• When GuardDuty is enabled, it may generate the Malware Protection for S3 finding type to indicate the presence of malware in the scanned S3 object. • You can potentially check the S3 object scan result by using one or more options under Monitoring S3 object scans in Malware Protection for S3. These include using Amazon EventBridge, CloudWatch metrics for Malware Protection plan, and tagging scanned objects.	You can potentially check the S3 object scan result by using one or more options under Monitoring S3 object scans in Malware Protection for S3. These include using Amazon EventBridge, CloudWatch metrics for Malware Protection plan, and tagging scanned objects.

Using tag-based access control (TBAC) with Malware Protection for S3

When enabling Malware Protection for S3 for your bucket, you can optionally choose to enable tagging. After attempting to scan a newly uploaded S3 object in the selected bucket, GuardDuty adds a tag to the scanned object to provide the malware scan status. There is a direct usage cost associated when you enable tagging. For more information, see Pricing and usage cost for Malware Protection for S3.

GuardDuty uses a predefined tag with the key as `GuardDutyMalwareScanStatus` and the value as one of the malware scan statuses. For information about these values, see the section called “S3 object potential scan status and result status”.

Considerations for GuardDuty to add a tag to your S3 object:

- By default, you can associate up to 10 tags with an object. For more information, see Categorizing your storage using tags in the *Amazon S3 User Guide*.

  If all 10 tags are already in use, GuardDuty can't add the predefined tag to the scanned object. GuardDuty also publishes the scan result to your default EventBridge event bus. For more information, see Monitoring S3 object scans with Amazon EventBridge.
- When the selected IAM role doesn't include the permission for GuardDuty to tag the S3 object, then even with tagging enabled for your protected bucket, GuardDuty will be unable to add tag to this scanned S3 object. For more information about the required IAM role permission for tagging, see Create or update IAM role policy.

  GuardDuty also publishes the scan result to your default EventBridge event bus. For more information, see Monitoring S3 object scans with Amazon EventBridge.

Adding TBAC on S3 bucket resource

You can use the S3 bucket resource policies to manage tag-based access control (TBAC) for your S3 objects. You can provide access to specific users to access and read the S3 object. If you have an organization that was created by using AWS Organizations, you must enforce that no one can modify the tags added by GuardDuty. For more information, see Preventing tags from being modified except by authorized principals in the *AWS Organizations User Guide*. The example used in the linked topic mentions `ec2`. When you use this example, replace *`ec2`* with `s3`.

The following list explains what you can do by using TBAC:

- Prevent all the users except Malware Protection for S3 service principal from reading the S3 objects that are not yet tagged with the following tag key-value pair:

  `GuardDutyMalwareScanStatus:`*`Potential key value`*
- Allow only GuardDuty to add the tag key `GuardDutyMalwareScanStatus` with value as the scan result, to a scanned S3 object. The following policy template can allow specific users that have access, to potentially override the tag key-value pair.

Example S3 bucket resource policy:

Replace the following placeholder values in the example policy:

- *IAM-role-name* - Provide the IAM role that you used for configuring Malware Protection for S3 in your bucket.
- *555555555555* - Provide the AWS account associated with the protected bucket.
- *amzn-s3-demo-bucket* - Provide the protected bucket name.

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [{
            "Sid": "NoReadUnlessClean",
            "Effect": "Deny",
            "NotPrincipal": {
                "AWS": [
                "arn:aws:sts::555555555555:assumed-role/IAM-role-name/
GuardDutyMalwareProtection",
                "arn:aws:iam::555555555555:role/IAM-role-name"
                ]
            },
            "Action": [
                "s3:GetObject",
                "s3:GetObjectVersion"
            ],
            "Resource": "arn:aws:s3:::amzn-s3-demo-bucket/*",
            "Condition": {
                "StringNotEquals": {
                    "s3:ExistingObjectTag/GuardDutyMalwareScanStatus":
 "NO_THREATS_FOUND"
                }
            }
        },
        {
            "Sid": "OnlyGuardDutyCanTagScanStatus",
            "Effect": "Deny",
            "NotPrincipal": {
                "AWS": [
                    "arn:aws:sts::555555555555:assumed-role/IAM-role-name/
GuardDutyMalwareProtection",
                    "arn:aws:iam::555555555555:role/IAM-role-name"
                ]
            },
```

```
                "Action": "s3:PutObjectTagging",
                "Resource": "arn:aws:s3:::amzn-s3-demo-bucket/*",
                "Condition": {
                    "ForAnyValue:StringEquals": {
                        "s3:RequestObjectTagKeys": "GuardDutyMalwareScanStatus"
                    }
                }
            }
        ]
}
```

For more information about tagging your S3 resource, Tagging and access control policies.

Viewing and understanding protected bucket status

After enabling Malware Protection for S3 for a bucket, the status indicates whether the feature is configured and functional as expected. This status is associated with a unique Malware Protection plan identifier (ID). GuardDuty creates this ID at the time of enabling the feature.

Use the following procedure to view the status of your protected bucket:

1. Sign in to the AWS Management Console and open the GuardDuty console at https://console.aws.amazon.com/guardduty/.
2. In the navigation pane, select **Malware Protection for S3**.
3. In the **Protected buckets** table, view the corresponding **Status** column for your **S3 bucket**.

The following table lists and describes status values associated with your Malware Protection plan resource. By understanding what these statuses mean for your protected bucket, you can better ensure that GuardDuty initiates an automatic malware scan when an object gets uploaded.

Status	Description
Active	Your S3 bucket has been configured with Malware Protection for S3 successfully. When the status is *Active*, changes to the IAM role (deletion or permissions modification) won't update the status to *Warning*

Status	Description
	or *Error*. We recommend monitoring the scan status continuously by using any of the methods described in Monitoring S3 object scans.
Warning*	Malware Protection for S3 is designed to not get impacted when a warning shows up. When GuardDuty notices a new S3 object, it will initiate a malware scan. After initiating the scan successfully, the **Status** column value may take a few minutes to change to **Active**. You will receive an EventBridge notification after the **Status** column value updates.
Error*	Your bucket is not protected. None of the malware scans associated with this S3 bucket will complete. There could be one or more potential root causes.

*For information about potential issues and the corresponding steps to resolve them, see Troubleshooting Malware Protection plan status.

Troubleshooting Malware Protection plan status

For any protected bucket, GuardDuty displays the **Status** based on the ranking. For example, if a protected bucket has issues under both **Error** and **Warning** categories, GuardDuty will first display the issue that is associated with the **Error** status.

The following list includes the errors and the warning for the Malware Protection plan status.

Errors

- EventBridge notification is disabled for this S3 bucket
- EventBridge managed rule to receive S3 bucket events is missing
- S3 bucket no longer exists

Warning

Unable to put test object

EventBridge notification is disabled for this S3 bucket

The associated status reason code is `EVENTBRIDGE_MANAGED_EVENTS_DELIVERY_DISABLED`.

Status detail

GuardDuty uses EventBridge to receive a notification when a new object gets uploaded to this S3 bucket. This permission is missing in your IAM role.

Steps to troubleshoot

Option 1: Add the following permission statement to your IAM role:

```
{
          "Sid": "AllowEnableS3EventBridgeEvents",
          "Effect": "Allow",
          "Action": [
             "s3:PutBucketNotification",
             "s3:GetBucketNotification"
             ],
          "Resource": [
             "arn:aws:s3:::amzn-s3-demo-bucket"
            ]
}
```

Replace *`amzn-s3-demo-bucket`* with your Amazon S3 bucket name.

Option 2: Enable EventBridge notification by using the Amazon S3 console

1. Open the Amazon S3 console at https://console.aws.amazon.com/s3/.
2. On the **Buckets** page, under **General purpose buckets** tab, select the bucket name associated with this error.
3. On this bucket page, choose the **Properties** tab.
4. Under the **Amazon EventBridge** section, select **Edit**.
5. On the **Edit Amazon EventBridge** page, for **Send notification to Amazon EventBridge for all events in this bucket**, select **On**.
6. Choose **Save changes**.

It may take a few minutes for the **Status** column value to change to **Active**.

EventBridge managed rule to receive S3 bucket events is missing

The associated status reason code is `EVENTBRIDGE_MANAGED_RULE_DISABLED`.

Status detail

The EventBridge managed rule permissions to manage the EventBridge rule setup is missing.

Steps to troubleshoot

Add the following permission statement to your IAM role:

```
{
          "Sid": "AllowManagedRuleToSendS3EventsToGuardDuty",
         "Effect": "Allow",
         "Action": [
                 "events:PutRule",
                 "events:DeleteRule",
                 "events:PutTargets",
                 "events:RemoveTargets"
             ],
         "Resource": [
            "arn:aws:events:*:*:rule/DO-NOT-DELETE-
AmazonGuardDutyMalwareProtectionS3*"
            ],
         "Condition": {
            "StringEquals": {
               "events:ManagedBy": "malware-protection-plan.guardduty.amazonaws.com"
               }
            }
}
```

It may take a few minutes for the **Status** column value to change to **Active**.

S3 bucket no longer exists

The associated status reason code is `PROTECTED_RESOURCE_DELETED`.

Status detail

This S3 bucket was deleted from your account and it no longer exists.

Step to troubleshoot

If deleting the S3 bucket was not intentional, then you can create a new bucket by using the Amazon S3 console.

After creating the bucket successfully, enable Malware Protection for S3 by following the steps under the Configuring Malware Protection for S3 for your bucket page.

Unable to put test object

The associated status reason code is `INSUFFICIENT_TEST_OBJECT_PERMISSIONS`.

> **Note**
>
> The permission to add a test object is optional. Missing this permission in your IAM role doesn't prevent Malware Protection for S3 to initiate malware scan on a newly uploaded object. After a scan initiates successfully, it may take a few minutes for the Malware Protection plan **Status** to change from **Warning** to **Active**.
> If the IAM role includes this permission already, then this warning indicates a restrictive Amazon S3 bucket policy that does't allow the IAM access to put the test object in this S3 bucket.

Status detail

To validate the setup of the selected bucket, GuardDuty puts a test object in your bucket.

Steps to troubleshoot

You can choose to update the IAM role to include the missing permissions. To the selected IAM role, add the following permissions so that GuardDuty can put the test object to the selected resource:

```
{
         "Sid": "AllowPutValidationObject",
         "Effect": "Allow",
         "Action": [
           "s3:PutObject"
           ],
         "Resource": [
```

```
            "arn:aws:s3:::amzn-s3-demo-bucket/malware-protection-resource-validation-
object"
            ]
}
```

Replace *amzn-s3-demo-bucket* with your Amazon S3 bucket name. For information about IAM role permissions, see Create or update IAM role policy.

It may take a few minutes for the **Status** column value to change to **Active**.

Monitoring S3 object scans in Malware Protection for S3

When using Malware Protection for S3 with a GuardDuty detector ID, if your Amazon S3 object is potentially malicious, GuardDuty will generate Malware Protection for S3 finding type. Using the GuardDuty console and APIs, you can view the generated findings. For information about understanding this finding type, see Finding details.

When using Malware Protection for S3 without enabling GuardDuty (no detector ID), even when your scanned Amazon S3 object is potentially malicious, GuardDuty can't generate any findings.

Contents

- S3 object potential scan status and result status
- Monitoring S3 object scans with Amazon EventBridge
- Monitoring S3 object scans with GuardDuty managed tags
- S3 object scan status metrics in CloudWatch

S3 object potential scan status and result status

This section explains the potential S3 object scan status values and the scan result values.

An S3 object scan status indicates the status of the malware scan, such as completed, skipped, or failed.

An S3 object malware scan result status indicates the result of the scan based on the scan status value. Each malware scan result status value maps to a scan status.

The following list provides the potential S3 object scan result values. If you have enabled tagging, you can monitor the scan result by Using S3 Object Tags. After the scan, the tag value will have one of the following scan result values.

S3 object potential malware scan result status values

- NO_THREATS_FOUND – GuardDuty detected no potential threat associated with the scanned object.
- THREATS_FOUND – GuardDuty detected a potential threat associated with the scanned object.
- UNSUPPORTED – There are a few reasons why Malware Protection for S3 will skip a scan. Potential reasons include password-protected file, archives with extremely high compression ratios, Malware Protection for S3 quotas, and support for certain Amazon S3 features may be unavailable. For more information, see Capabilities of Malware Protection for S3.
- ACCESS_DENIED – GuardDuty can't access this object for scanning. Check the IAM role permissions associated with this bucket. For more information, see Create or update IAM role policy.

  If you have enabled post-scan S3 object tagging, see Troubleshooting S3 object post-scan tag failures.
- FAILED – GuardDuty can't perform malware scan on this object because of an internal error.

The following list provides potential S3 object scan status values and their mapping to the S3 object scan result.

S3 object potential scan status values

- **Completed** – The scan completed successfully and indicates whether the S3 object has malware. In this case, the potential S3 object scan result value could be either THREATS_FOUND or NO_THREATS_FOUND.
- **Skipped** – GuardDuty skips a malware scan when scanning this S3 object is not supported by Malware Protection for S3, or GuardDuty doesn't have access to the uploaded S3 object in the selected bucket.

  In this case, the potential S3 object scan result value could be either UNSUPPORTED or ACCESS_DENIED.

  GuardDuty will also skip the scan if the required IAM role gets deleted.

- **Failed** – Similar to the S3 object scan result value `FAILED`, this scan status means that GuardDuty was unable to perform malware scan on the S3 object because of an internal error.

Monitoring S3 object scans with Amazon EventBridge

Amazon EventBridge is a serverless event bus service that makes it easy to connect your applications with data from a variety of sources. EventBridge delivers a stream of real-time data from your own applications, Software-as-a-Service (SaaS) applications, and AWS services and routes that data to targets such as Lambda. This enables you to monitor events that happen in services, and build event-driven architectures. For more information, see the Amazon EventBridge User Guide.

As the owner account of an S3 bucket that is protected with Malware Protection for S3, GuardDuty publishes EventBridge notifications to the default event bus in the following scenarios:

- **Malware Protection plan resource status** changes for any of your protected buckets. For information about various statuses, see Viewing and understanding protected bucket status.

  For setting up Amazon EventBridge (EventBridge) rule for the resource status, see Malware Protection plan resource status.
- The **S3 object scan result** gets published to your default EventBridge event bus.

  The `s3Throttled` field indicates whether or not there was a delay in uploading or retrieving storage from Amazon S3. The value `true` indicates that there was a delay, and `false` indicates that there was no delay.

  If `s3Throttled` is `true` for your scan result, then Amazon S3 recommends setting up prefixes in a way that helps you reduce the transactions per second (TPS) for each prefix. For more information, see Best practices design patterns: optimizing Amazon S3 performance in the *Amazon S3 User Guide*.

  For setting up Amazon EventBridge (EventBridge) rule for the S3 object scan results, see S3 object scan result.
- There is a **post-scan tag failure event** because of the following reasons:
  - Your IAM role is missing permissions to tag the object.

    The Adding IAM policy permissions template includes the permission for GuardDuty to tag an object.

- The bucket resource or object specified in the IAM role no longer exists.
- The associated S3 object has already reached the maximum tag limit. For more information about the tag limit, see Categorizing your storage using tags in the *Amazon S3 User Guide*.

For setting up Amazon EventBridge (EventBridge) rule for the post-scan tag failure events, see Post-scan tag failure events.

Set up EventBridge rules

You can set up EventBridge rules in your account to send either resource status, post-scan tag failure events, or the S3 object scan result to another AWS service. As a delegated GuardDuty administrator account, you will receive the Malware Protection plan resource status notification when there is a change in the status.

Standard EventBridge pricing will apply. For more information, see Amazon EventBridge pricing.

All the values that show up in *`red`* are placeholders for the example. These values will change based on the values in your account, and whether or not malware is detected.

Topics

- Malware Protection plan resource status
- S3 object scan result
- Post-scan tag failure events

Malware Protection plan resource status

You can create an EventBridge event pattern based on the following scenarios:

Potential `detail-type` values

- `"GuardDuty Malware Protection Resource Status Active"`
- `"GuardDuty Malware Protection Resource Status Warning"`
- `"GuardDuty Malware Protection Resource Status Error"`

Event pattern

```
{
```

```
        "detail-type": ["potential detail-type"],
        "source": ["aws.guardduty"]
}
```

Sample notification schema for `GuardDuty Malware Protection Resource Status Active`:

```
{
    "version": "0",
    "id": "6a7e8feb-b491-4cf7-a9f1-bf3703467718",
    "detail-type": "GuardDuty Malware Protection Resource Status Active",
    "source": "aws.guardduty",
    "account": "111122223333",
    "time": "2017-12-22T18:43:48Z",
    "region": "us-east-1",
    "resources": ["arn:aws:guardduty:us-east-1:111122223333:malware-protection-plan/
b4c7f464ab3a4EXAMPLE"],
    "detail": {
        "schemaVersion": "1.0",
        "eventTime": "2024-02-28T01:01:01Z",
        "s3BucketDetails": {
            "bucketName": "amzn-s3-demo-bucket"
        },
        "resourceStatus": "ACTIVE"
    }
}
```

Sample notification schema for `GuardDuty Malware Protection Resource Status Warning`:

```
{
    "version": "0",
    "id": "6a7e8feb-b491-4cf7-a9f1-bf3703467718",
    "detail-type": "GuardDuty Malware Protection Resource Status warning",
    "source": "aws.guardduty",
    "account": "111122223333",
    "time": "2017-12-22T18:43:48Z",
    "region": "us-east-1",
    "resources": ["arn:aws:guardduty:us-east-1:111122223333:malware-protection-plan/
b4c7f464ab3a4EXAMPLE"],
    "detail": {
        "schemaVersion": "1.0",
```

```
        "eventTime": "2024-02-28T01:01:01Z",
        "s3BucketDetails": {
            "bucketName": "amzn-s3-demo-bucket"
        },
        "resourceStatus": "WARNING",
        "statusReasons": [
         {
            "code": "INSUFFICIENT_TEST_OBJECT_PERMISSIONS"
         }
        ]
    }
}
```


Sample notification schema for `GuardDuty Malware Protection Resource Status Error`:


```
{
    "version": "0",
    "id": "fc7a35b7-83bd-3c1f-ecfa-1b8de9e7f7d2",
    "detail-type": "GuardDuty Malware Protection Resource Status Error",
    "source": "aws.guardduty",
    "account": "111122223333",
    "time": "2017-12-22T18:43:48Z",
    "region": "us-east-1",
    "resources": ["arn:aws:guardduty:us-east-1:111122223333:malware-protection-plan/
b4c7f464ab3a4EXAMPLE"],
    "detail": {
        "schemaVersion": "1.0",
        "eventTime": "2024-02-28T01:01:01Z",
        "s3BucketDetails": {
            "bucketName": "amzn-s3-demo-bucket"
        },
        "resourceStatus": "ERROR",
        "statusReasons": [
        {
            "code": "EVENTBRIDGE_MANAGED_EVENTS_DELIVERY_DISABLED"
        }
       ]
    }
}
```


Based on the reason behind the `resourceStatus` ERROR, the `statusReasons` value will get populated.

For information about troubleshooting steps for the following warning and errors, see Troubleshooting Malware Protection plan status.

S3 object scan result

```
{
  "detail-type": ["GuardDuty Malware Protection Object Scan Result"],
  "source": ["aws.guardduty"]
}
```

Sample notification schema for NO_THREATS_FOUND:


```
{
    "version": "0",
    "id": "72c7d362-737a-6dce-fc78-9e27a0171419",
    "detail-type": "GuardDuty Malware Protection Object Scan Result",
    "source": "aws.guardduty",
    "account": "111122223333",
    "time": "2024-02-28T01:01:01Z",
    "region": "us-east-1",
    "resources": ["arn:aws:guardduty:us-east-1:111122223333:malware-protection-plan/
b4c7f464ab3a4EXAMPLE"],
    "detail": {
        "schemaVersion": "1.0",
        "scanStatus": "COMPLETED",
        "resourceType": "S3_OBJECT",
        "s3ObjectDetails": {
            "bucketName": "amzn-s3-demo-bucket",
            "objectKey": "APKAEIBAERJR2EXAMPLE",
            "eTag": "ASIAI44QH8DHBEXAMPLE",
            "versionId" : "d41d8cd98f00b204e9800998eEXAMPLE",
            "s3Throttled": false
        },
        "scanResultDetails": {
            "scanResultStatus": "NO_THREATS_FOUND",
            "threats": null
        }
    }
}
```


Sample notification schema for THREATS_FOUND:

```
{
```

```
    "version": "0",
    "id": "72c7d362-737a-6dce-fc78-9e27a0171419",
    "detail-type": "GuardDuty Malware Protection Object Scan Result",
    "source": "aws.guardduty",
    "account": "111122223333",
    "time": "2024-02-28T01:01:01Z",
    "region": "us-east-1",
    "resources": ["arn:aws:guardduty:us-east-1:111122223333:malware-protection-plan/
b4c7f464ab3a4EXAMPLE"],
    "detail": {
        "schemaVersion": "1.0",
        "scanStatus": "COMPLETED",
        "resourceType": "S3_OBJECT",
        "s3ObjectDetails": {
            "bucketName": "amzn-s3-demo-bucket",
            "objectKey": "APKAEIBAERJR2EXAMPLE",
            "eTag": "ASIAI44QH8DHBEXAMPLE",
            "versionId" : "d41d8cd98f00b204e9800998eEXAMPLE",
            "s3Throttled": false
        },
        "scanResultDetails": {
            "scanResultStatus": "THREATS_FOUND",
            "threats": [
                {
                    "name": "EICAR-Test-File (not a virus)"
                }
            ]
        }
    }
}
```


> **Note**
>
> The `scanResultDetails.Threats` field contains only one threat. By default, the Malware Protection for S3 scan reports the first detected threat. After this, the `scanStatus` is set to `COMPLETED`.

Sample notification schema for scan result status `UNSUPPORTED` (Skipped):


```
{
    "version": "0",
```

```
    "id": "72c7d362-737a-6dce-fc78-9e27a0EXAMPLE",
    "detail-type": "GuardDuty Malware Protection Object Scan Result",
    "source": "aws.guardduty",
    "account": "111122223333",
    "time": "2024-02-28T01:01:01Z",
    "region": "us-east-1",
    "resources": ["arn:aws:guardduty:us-east-1:111122223333:malware-protection-plan/
b4c7f464ab3a4EXAMPLE"],
    "detail": {
        "schemaVersion": "1.0",
        "scanStatus": "SKIPPED",
        "resourceType": "S3_OBJECT",
        "s3ObjectDetails": {
            "bucketName": "amzn-s3-demo-bucket",
            "objectKey": "APKAEIBAERJR2EXAMPLE",
            "eTag": "ASIAI44QH8DHBEXAMPLE",
            "versionId" : "d41d8cd98f00b204e9800998eEXAMPLE",
            "s3Throttled": false
        },
        "scanResultDetails": {
            "scanResultStatus": "UNSUPPORTED",
            "threats": null
        }
    }
}
```

Sample notification schema for scan result status `ACCESS_DENIED` (Skipped):

```
{
    "version": "0",
    "id": "72c7d362-737a-6dce-fc78-9e27a0EXAMPLE",
    "detail-type": "GuardDuty Malware Protection Object Scan Result",
    "source": "aws.guardduty",
    "account": "111122223333",
    "time": "2024-02-28T01:01:01Z",
    "region": "us-east-1",
    "resources": ["arn:aws:guardduty:us-east-1:111122223333:malware-protection-plan/
b4c7f464ab3a4EXAMPLE"],
    "detail": {
        "schemaVersion": "1.0",
        "scanStatus": "SKIPPED",
        "resourceType": "S3_OBJECT",
        "s3ObjectDetails": {
```

```
            "bucketName": "amzn-s3-demo-bucket",
            "objectKey": "APKAEIBAERJR2EXAMPLE",
            "eTag": "ASIAI44QH8DHBEXAMPLE",
            "versionId" : "d41d8cd98f00b204e9800998eEXAMPLE",
            "s3Throttled": false
        },
        "scanResultDetails": {
            "scanResultStatus": "ACCESS_DENIED",
            "threats": null
        }
    }
}
```

Sample notification schema for scan result status `FAILED`:

```
{
    "version": "0",
    "id": "72c7d362-737a-6dce-fc78-9e27a0EXAMPLE",
    "detail-type": "GuardDuty Malware Protection Object Scan Result",
    "source": "aws.guardduty",
    "account": "111122223333",
    "time": "2024-02-28T01:01:01Z",
    "region": "us-east-1",
    "resources": ["arn:aws:guardduty:us-east-1:111122223333:malware-protection-plan/
b4c7f464ab3a4EXAMPLE"],
    "detail": {
        "schemaVersion": "1.0",
        "scanStatus": "FAILED",
        "resourceType": "S3_OBJECT",
        "s3ObjectDetails": {
            "bucketName": "amzn-s3-demo-bucket",
            "objectKey": "APKAEIBAERJR2EXAMPLE",
            "eTag": "ASIAI44QH8DHBEXAMPLE",
            "versionId" : "d41d8cd98f00b204e9800998eEXAMPLE",
            "s3Throttled": false
        },
        "scanResultDetails": {
            "scanResultStatus": "FAILED",
            "threats": null
        }
    }
}
```

Post-scan tag failure events

Event pattern:

```
{
      "detail-type": "GuardDuty Malware Protection Post Scan Action Failed",
      "source": "aws.guardduty"
 }
```

Sample notification schema for ACCESS_DENIED:


```
{
    "version": "0",
    "id": "746acd83-d75c-5b84-91d2-dad5f13ba0d7",
    "detail-type": "GuardDuty Malware Protection Post Scan Action Failed",
    "source": "aws.guardduty",
    "account": "111122223333",
    "time": "2024-06-10T16:16:08Z",
    "region": "us-east-1",
    "resources": ["arn:aws:guardduty:us-east-1:111122223333:malware-protection-plan/
b4c7f464ab3a4EXAMPLE"],
    "detail": {
        "schemaVersion": "1.0",
        "eventTime": "2024-06-10T16:16:08Z",
        "s3ObjectDetails": {
            "bucketName": "amzn-s3-demo-bucket",
            "objectKey": "2024-03-10-16-16-00-7D723DE8DBE9Y2E0",
            "eTag": "0e9eeec810ad8b61d69112c15c2a5hb6",
            "versionId" : "d41d8cd98f00b204e9800998eEXAMPLE",
            "s3Throttled": false
        },
        "postScanActions": [{
            "actionType": "TAGGING",
            "failureReason": "ACCESS_DENIED"
        }]
    }
}
```


Sample notification schema for MAX_TAG_LIMIT_EXCEEDED:


```
{
    "version": "0",
    "id": "746acd83-d75c-5b84-91d2-dad5f13ba0d7",
```

```
    "detail-type": "GuardDuty Malware Protection Post Scan Action Failed",
    "source": "aws.guardduty",
    "account": "111122223333",
    "time": "2024-06-10T16:16:08Z",
    "region": "us-east-1",
    "resources": ["arn:aws:guardduty:us-east-1:111122223333:malware-protection-plan/
b4c7f464ab3a4EXAMPLE"],
    "detail": {
        "schemaVersion": "1.0",
        "eventTime": "2024-06-10T16:16:08Z",
        "s3ObjectDetails": {
            "bucketName": "amzn-s3-demo-bucket",
            "objectKey": "2024-03-10-16-16-00-7D723DE8DBE9Y2E0",
            "eTag": "0e9eeec810ad8b61d69112c15c2a5hb6",
            "versionId" : "d41d8cd98f00b204e9800998eEXAMPLE",
            "s3Throttled": false
        },
        "postScanActions": [{
            "actionType": "TAGGING",
            "failureReason": "MAX_TAG_LIMIT_EXCEEDED"
        }]
    }
}
```

To troubleshoot these failure reasons, see Troubleshooting S3 object post-scan tag failures.

Monitoring S3 object scans with GuardDuty managed tags

Use enable tagging option so that GuardDuty can add tags to your Amazon S3 object after completing the malware scan.

Considerations for enabling tagging

- There is an associated usage cost when GuardDuty tags your S3 objects. For more information, see Pricing and usage cost for Malware Protection for S3.
- You must keep the required tagging permissions to your preferred IAM role associated with this bucket; otherwise, GuardDuty can't add tags to your scanned objects. The IAM role already includes the permissions to add tags to the scanned S3 objects. For more information, see Create or update IAM role policy.
- By default, you can associate up to 10 tags with an S3 object. For more information, see Using tag-based access control (TBAC).

After you enable tagging for an S3 bucket or specific prefixes, any newly uploaded object that gets scanned, will have an associated tag in the following key-value pair format:

`GuardDutyMalwareScanStatus:`*`Scan-Result-Status`*

For information about potential tag values, see S3 object potential scan status and result status.

Troubleshooting S3 object post-scan tag failures in Malware Protection for S3

This section applies to you only if you Enable tagging for scanned objects in your protected bucket.

When GuardDuty attempts to add a tag to your scanned S3 object, the action of tagging may result in a failure. The potential reasons why this may happen to your bucket are `ACCESS_DENIED` and `MAX_TAG_LIMIT_EXCEEDED`. Use the following topics to understand the potential reasons for these post-scan tag failure reasons and troubleshoot them.

ACCESS_DENIED

The following list provides potential reasons that may cause this issue:

- The IAM role used for this protected S3 bucket is missing the **AllowPostScanTag** permission. Verify that the associated IAM role uses this bucket policy. For more information, see Create or update IAM role policy.
- The protected S3 bucket policy does't allow GuardDuty to add tags to this object.
- The scanned S3 object no longer exists.

MAX_TAG_LIMIT_EXCEEDED

By default, you can associate up to 10 tags with an S3 object. For more information, see Considerations for GuardDuty to add a tag to your S3 object under Enable tagging for scanned objects.

S3 object scan status metrics in CloudWatch

You can monitor GuardDuty using CloudWatch, which collects raw data and processes it into readable, near real-time metrics. These statistics are retained for 15 months, so that you can access historical information and gain a better perspective on how Malware Protection for S3 is performing. You can also set alarms that watch for certain thresholds, and send notifications or take actions when those thresholds are met. For more information, see the Amazon CloudWatch User Guide.

The CloudWatch metrics for Malware Protection for S3 are available at the resource level. You can query these metrics for each protected resource separately. The metrics are reported in the `AWS/GuardDuty/MalwareProtection` namespace. You can set up alarms on specific resources to monitor security posture.

Malware scan status metrics

Metric	Description
`CompletedScanCount`	The number of S3 object malware scans that completed in a given time frame. Valid Dimensions: • `Malware Protection Plan Id` `Resource Name` Units: Count
`FailedScanCount`	The number of S3 object malware scans that failed in a given time frame. **Valid Dimensions**: • `Malware Protection Plan Id` `Resource Name` Units: Count
`SkippedScanCount`	The number of S3 object malware scans that were skipped in a given time frame. **Valid Dimensions**: • `Malware Protection Plan Id` `Resource Name`

	`Skipped Reason` **Potential values** • `Unsupported` • `MissingPermissions` Units: Count

Malware scan result metrics

`InfectedScanCount`	The number of S3 object malware scans that detected potentially malicious object in a given time frame. Valid Dimensions: • `Malware Protection Plan Id` `Resource Name` Units: Count
`CompletedScanBytes`	The number of S3 object bytes scanned in a given time frame. Valid Dimensions: • `Malware Protection Plan Id` `Resource Name` Units: Count

> **Note**
>
> By default, the statistics in the CloudWatch metrics are AVG.

The following dimensions are supported for the Malware Protection for S3 metrics.

Dimension	Description
`Malware Protection Plan Id`	The unique identifier that is associated with the Malware Protection plan resource that GuardDuty creates for your protected resource.
`Resource Name`	The name of the protected resource.
`Skipped Reason`	The reason why an S3 object malware scan was skipped. **Potential values** • `Unsupported` • `MissingPermissions`

For information about accessing and querying these metrics, see Use Amazon CloudWatch metrics in the *Amazon CloudWatch User Guide*.

For information about setting up alarms, see Using Amazon CloudWatch alarms in the *Amazon CloudWatch User Guide*.

Editing Malware Protection plan for a protected bucket

You may need to edit the preferred IAM permissions policy, enable or disable tagging of the scanned S3 object, or add or remove S3 object prefixes. For example, when you enabled Malware Protection for S3 for your bucket, you decided to not enable tagging the scanned S3 object with the scan result. However, now you want GuardDuty to add the predefined tag and the scan result as the tag value.

Choose a preferred access method to update the Malware Protection plan for your protected S3 bucket.

Console

To edit a Malware Protection plan

1. Sign in to the AWS Management Console and open the GuardDuty console at https://console.aws.amazon.com/guardduty/.
2. In the navigation pane, choose **Malware Protection for S3**.
3. Under **Protected buckets**, select the bucket for which you want to edit the existing configuration.
4. Choose **Edit**.
5. Update the existing configuration and settings for your bucket and confirm the changes. For information about description and steps for each section, see Enabling Malware Protection for S3 for your bucket.

   Monitor the **Status** column for this protected bucket. If it appears as either **Warning** or **Error**, see Troubleshooting Malware Protection plan status.

API/CLI

To edit Malware Protection plan by using API or AWS CLI

- **By using API**

  Run the UpdateMalwareProtectionPlan API by using the Malware Protection plan ID associated with this plan resource.

  To retrieve the Malware Protection plan ID in a specific Region, you can run the ListMalwareProtectionPlans API in that Region.
- **By using AWS CLI**

  The following list provides AWS CLI example commands to update the Malware Protection plan resource. You will need the Malware Protection plan ID associated with your S3 bucket.

  AWS CLI example commands

  - Use the following AWS CLI command to **enable or disable** tagging for the Malware Protection plan resource associated with your S3 bucket:

```
aws guardduty update-malware-protection-plan --malware-protection-plan-
id 4cc8bf26c4d75EXAMPLE --actions "Tagging"={"Status"="ENABLED|DISABLED"}
```

- Use the following AWS CLI command to **add an object prefix** to the Malware Protection plan resource associated with your S3 bucket:

```
aws guardduty update-malware-protection-plan --malware-
protection-plan-id 4cc8bf26c4d75EXAMPLE --protected-resource
 "S3Bucket"={"ObjectPrefixes"=["amzn-s3-demo-1", "amzn-s3-demo-2"]}
```

  Make sure to include the existing object prefixes in this command; otherwise, GuardDuty will remove those prefixes when editing the Malware Protection plan resource.

- Use the following AWS CLI command to **remove an object prefix** from the Malware Protection plan resource associated with your S3 bucket:

```
aws guardduty update-malware-protection-plan --malware-protection-plan-
id 4cc8bf26c4d75EXAMPLE --protected-resource "S3Bucket"={"ObjectPrefixes"=[""]}
```

If you don't already have the Malware Protection plan ID for this resource, you can run the following AWS CLI command and replace *us-east-1* with the Region for which you want to list the Malware Protection plan IDs.

```
aws guardduty list-malware-protection-plans --region us-east-1
```

Disabling Malware Protection for S3 for a protected bucket

When you disable Malware Protection for S3 for a protected bucket, GuardDuty deletes the Malware Protection plan ID associated with that bucket. GuardDuty will no longer start a malware scan when a new object gets uploaded to this bucket or one of the selected object prefixes.

If you have enabled GuardDuty and now want to suspend or disable GuardDuty, see Suspending or disabling GuardDuty. Because there is no concept of detector ID in Malware Protection for S3, disabling or suspending GuardDuty **doesn't** impact the status of a protected bucket in your account. You can continue using Malware Protection for S3 feature independently with the associated standard pricing. For more information, see Reviewing usage cost for Malware Protection for S3. To stop using Malware Protection for S3, you will need to disable it for all the protected buckets in your account. If you want to continue using GuardDuty and disable only Malware Protection for S3 for a bucket, the following steps are not going to impact the configuration of the GuardDuty service and other protection plans that you may have enabled.

Choose a preferred access method to disable Malware Protection for S3 in your protected S3 bucket.

Console

To disable Malware Protection for S3 by using GuardDuty console

1. Sign in to the AWS Management Console and open the GuardDuty console at https://console.aws.amazon.com/guardduty/.
2. In the navigation pane, choose **Malware Protection for S3**.
3. Under **Protected buckets**, select the bucket for which you want to disable Malware Protection for S3.

   You can select only one protected bucket at a time. To disable Malware Protection for S3 for more than one bucket, follow these steps again for another S3 bucket.
4. Choose **Disable** to confirm the selection.

API/CLI

To disable Malware Protection for S3 by using API or AWS CLI

- **By using API**

  Run the DeleteMalwareProtectionPlan API by using the Malware Protection plan ID associated with this plan resource.

  To retrieve the Malware Protection plan ID, you can run the ListMalwareProtectionPlans API.
- **By using AWS CLI**

  Alternatively, you can run the following AWS CLI command to disable Malware Protection for S3 by replacing *4cc8bf26c4d75EXAMPLE* with the Malware Protection plan ID associated to this S3 bucket:

  ```
  aws guardduty delete-malware-protection-plan --malware-protection-plan-
  id 4cc8bf26c4d75EXAMPLE
  ```

If you don't already have the Malware Protection plan ID for this S3 bucket, you can run the following AWS CLI command and replace *us-east-1* with the Region for which you want to list the Malware Protection plan IDs.

```
aws guardduty list-malware-protection-plans --region us-east-1
```

Supportability of Amazon S3 features

The following table specifies whether or not Malware Protection for S3 supports the listed Amazon S3 features.

Is the support available?	Description
Yes	S3 objects can be retrieved without restoring asynchron ously.

Is the support available?	Description
Conditional	• Intelligent Tiering support is available for S3 objects in the Frequent, Infrequent, and Archive Instance Access tiers. • The opt-in Archive and Deep Archive tiers are not supported. • Intelligent Tiering always creates a new object in Frequent Access tier. Therefore, object scan on create is supported. • Future Intelligent tiering features might start out objects in Archive. Therefore, this is not supported.

Is the support available?	Description
No	GuardDuty supports only general purpose buckets for Malware Protection for S3.
No	The S3 objects must be restored before they can be accessed.

Is the support available?	Description
No	Malware Protection for S3 is not supported on Outposts.
Yes	All the uploaded S3 objects are scanned for malware. If you uploaded an object with file version v1 and immediately uploaded another version override with v2, then GuardDuty will scan both the object file versions v1 and v2. However, the scan start time might not be in the same order.
Yes	If the destination bucket is a protected resource, then GuardDuty will scan all the S3 objects are replicated to the prefixes that are protected and monitored.
No	You can't define a replication rule based on the scan result tag. Amazon S3 does't support replication for tag, except for on create.

Is the support available?	Description
Yes	GuardDuty supports malware scans for S3 objects that are encrypted with managed and customer managed keys. Ensure that the IAM role includes the permission to use the key. For more information, see Adding IAM policy permissions.

Is the support available?	Description
No	Malware Protection for S3 doesn't support scanning S3 objects that are encrypted with keys that are not accessible.
No	When your Amazon S3 objects are encrypted by using Amazon S3 Encryption Client, your objects aren't exposed to any third party, including AWS. For information on why this is not supported, see Protecting data by using client-side encryption. **Note** CSE-KMS encrypted objects are received as an encrypted blob where the encryption can't be determined. Therefore, GuardDuty processes them as they are received, and scans the encrypted blob as a regular file. GuardDuty doesn't return an `UNSUPPORTED` scan status for such objects, unless any of the Quotas in Malware Protection for S3 exceeds.
Yes	Locked S3 objects are locked based on WORM - Write Once Read Many. Malware Protection for S3 can access and scan the objects.

Is the support available?	Description
Yes	Malware Protection for S3 can scan the buckets that are set up with *Requester Pays*. The requester will pay for the S3 calls. For more information, see Using Requester Pays buckets for storage transfers and usage in the *Amazon S3 User Guide*.
Yes	You can define lifecycle policies based on the scan result tag. For example, auto-delete malicious objects. For more information about lifcycle configuration, see Managing your storage lifecycle in the *Amazon S3 User Guide*.
Yes	You can define bucket resource policies based on your S3 object scan result tag. For example, prevent access to S3 objects that are not yet scanned, or GuardDuty detected threats. For more information, see Using tag-based access control (TBAC) with Malware Protection for S3.

Quotas in Malware Protection for S3

This section provides default quotas, often referred to as limits. Unless specified, each quota is Region-specific. To view default quotas specific to using the foundational GuardDuty service, see Amazon GuardDuty quotas.

The following tables describe the multiple quotas that will apply to your AWS account.

AWS default quota value	Is it adjustable?	Description
100 GB	No	The maximum S3 object size that GuardDuty will attempt to scan for malware.

AWS default quota value	Is it adjustable?	Description
		Although this quota is not adjustabl e, if you need to scan larger objects, contact AWS Support to determine if GuardDuty can increase the quota for your use case.
100 GB	No	The maximum amount of data that GuardDuty can extract and analyze from an archive file. GuardDuty will skip archive files extracting to more than 100 GB.
10,000	No	The maximum number of files that GuardDuty can extract and analyze in an archive file. If the archive contains more than 10,000 files, then GuardDuty will have to skip the archived file. **Note** Compound files types are potentially subject to these limits. The file types include, but are not limited to, Multipurpose Internet Mail Extensions (MIME) encoded email messages, Compiled Python (PYC) files, Compiled HTML Help (CHM) files, all installers, and OpenDocument Format (ODF) documents.

AWS default quota value	Is it adjustable?	Description
5	No	The maximum levels of nested archives that GuardDuty can extract. If the archive includes files that are nested beyond this value, then GuardDuty will skip those nested files.
25	No	The maximum number of S3 buckets for which you can enable Malware Protection for S3. This quota limit is per account in each Region.

GuardDuty RDS Protection

RDS Protection in Amazon GuardDuty analyzes and profiles RDS login activity for potential access threats to your Amazon Aurora databases (Amazon Aurora MySQL-Compatible Edition and Aurora PostgreSQL-Compatible Edition) and Amazon RDS for PostgreSQL.

RDS Protection helps you identify potentially suspicious login behavior on these supported databases. GuardDuty continuously monitors and profiles RDS login activity for anomalous activity. For example, a previously unseen external actor has unauthorized access to your database, or adversary attempts brute-force access by guessing the database's password.

With the launch of Amazon Aurora PostgreSQL Limitless Database, GuardDuty expands RDS Protection to now also support monitoring login activity from Limitless Databases. For AWS accounts that have already enabled RDS Protection, GuardDuty will automatically start monitoring login data from their Limitless Databases. For accounts that have not yet enabled RDS Protection, you can learn more about the 30-day free trial and choose to enable this feature. To enable this feature, see Enabling RDS Protection in multiple-account environments or Enabling RDS Protection for a standalone account.

> **Note**
>
> RDS for PostgreSQL read replica instances require the primary database instance to be on a supported database version, and to be successfully replicated from primary database. For information about read replicas, see Working with DB instance read replicas in *Amazon RDS User Guide*.

RDS Protection doesn't require additional infrastructure; it is designed so as not to affect the performance of your database instances. When RDS Protection detects a potentially suspicious or m anomalous login attempt, GuardDuty generates one or more RDS Protection finding types with details about the potentially compromised database.

30-day free trial

- When you enable GuardDuty in an AWS account in a new Region for the first time, you get a 30-day free trial. In this case, GuardDuty will also enable RDS Protection, which is included in the free trial. RDS Protection will start monitoring the login behavior of your database.

- When you are already using GuardDuty and decide to enable RDS Protection in a new Region for the first time, your account in this Region will get a 30-day free trial for RDS Protection.
- If you have already enabled RDS Protection, then with the launch of Amazon Aurora PostgreSQL Limitless Database, GuardDuty will automatically start monitoring login activity for the Limitless Databases. If your RDS Protection 30-day free trial has expired already, then you will start incurring usage costs related to monitoring of Limitless Databases.
- You can choose to disable RDS Protection in any Region at any time.
- During the 30-day free trial, you can get an estimate of your usage costs in that account and Region. After the 30-day free trial ends, RDS Protection doesn't get disabled automatically. Your account in this Region will start incurring usage cost. For more information, see Estimating GuardDuty usage cost.

When the RDS Protection feature is not enabled, GuardDuty does't detect anomalous or suspicious login behavior. If you disable RDS Protection, GuardDuty immediately stops monitoring RDS login activity, and will not detect any potential threat to your supported database instances or generate associated finding types.

For AWS Regions where Aurora PostgreSQL Limitless Databases are supported, see Requirements for Aurora PostgreSQL Limitless Database.

Supported Amazon Aurora, Amazon RDS, and Aurora Limitless databases

The following table shows the supported Aurora and Amazon RDS database versions for RDS Protection.

Amazon Aurora and Amazon RDS DB engine	Supported engine versions
Aurora MySQL	• 2.10.2 or later • 3.02.1 or later
Aurora PostgreSQL	• 10.23 or later • 11.12 or later • 12.7 or later • 13.3 or later

Amazon Aurora and Amazon RDS DB engine	Supported engine versions
	• 14.3 or later • 15.2 or later • 16.1 or later
RDS for PostgreSQL	• 14.5 or later • 13.8 or later • 12.12 or later • 11.17 or later • RDS for PostgreSQL version 15 • RDS for PostgreSQL version 16
Amazon Aurora PostgreSQL Limitless Database	`16.4-limitless`

RDS login activity

When you enable the RDS Protection feature, GuardDuty automatically starts monitoring RDS login activity for your databases, directly from the Aurora and Amazon RDS services. RDS login activity captures both successful and failed login attempts made to the Supported Amazon Aurora, Amazon RDS, and Aurora Limitless databases in your AWS environment. If there is an indication of anomalous login behavior, GuardDuty generates a finding with details about the potentially compromised database. When you enable RDS Protection for the first time or you have a newly created database instance, there is a learning period to baseline normal behavior. For this reason, newly enabled or newly created database instances may not have an associated anomalous login finding for up to two weeks.

When RDS Protection detects a potential threat, such as an unusual pattern in a series of successful, failed, or incomplete login attempts, GuardDuty generates one or more RDS Protection finding types. Based on the finding type, it may include details about the anomalous behavior, such as RDS login activity-based anomalies.

GuardDuty doesn't manage your Supported databases or RDS login activity, or make RDS login activity available to you.

Enabling RDS Protection in multiple-account environments

In a multiple-account environment, only the delegated GuardDuty administrator account has the option to enable or disable the RDS Protection feature for the member accounts in their organization. The GuardDuty member accounts can't modify this configuration from their accounts. The delegated GuardDuty administrator account manages their member accounts using AWS Organizations. This delegated GuardDuty administrator account can choose to auto-enable RDS login activity monitoring for all the new accounts as they join the organization. For more information about multiple-account environments, see [Multiple accounts in GuardDuty](#).

Enabling RDS Protection for delegated GuardDuty administrator account

Choose your preferred access method to configure RDS Login Activity Monitoring for the delegated GuardDuty administrator account.

Console

1. Open the GuardDuty console at https://console.aws.amazon.com/guardduty/.
2. In the navigation pane, choose **RDS Protection**.
3. On the **RDS Protection** page, choose **Edit**.
4. Do one of the following:

   Using Enable for all accounts

   - Choose **Enable for all accounts**. This will enable the protection plan for all the active GuardDuty accounts in your AWS organization, including the new accounts that join the organization.
   - Choose **Save**.

   Using Configure accounts manually

   - To enable the protection plan only for the delegated GuardDuty administrator account account, choose **Configure accounts manually**.
   - Choose **Enable** under the **delegated GuardDuty administrator account (this account)** section.
   - Choose **Save**.

API/CLI

Run the updateDetector API operation using your own regional detector ID and passing the `features` object name as `RDS_LOGIN_EVENTS` and `status` as `ENABLED`.

Alternatively, you can use AWS CLI to enable RDS Protection. Run the following command, and replace *12abc34d567e8fa901bc2d34e56789f0* with your account's detector ID and *us-east-1* with the Region where you want to enable RDS Protection.

To find the `detectorId` for your account and current Region, see the **Settings** page in the https://console.aws.amazon.com/guardduty/ console, or run the ListDetectors API.

```
aws guardduty update-detector --detector-id 12abc34d567e8fa901bc2d34e56789f0 --
region us-east-1 --features '[{"Name": "RDS_LOGIN_EVENTS", "Status": "ENABLED"}]'
```

Auto-enable RDS Protection for all member accounts

Choose your preferred access method to enable the RDS Protection feature for all member accounts. This includes existing member accounts and the new accounts that join the organization.

Console

1. Open the GuardDuty console at https://console.aws.amazon.com/guardduty/.

   Make sure to use the delegated GuardDuty administrator account credentials.

2. Do one of the following:

   Using the RDS Protection page

   1. In the navigation pane, choose **RDS Protection**.
   2. Choose **Enable for all accounts**. This action automatically enables RDS Protection for both existing and new accounts in the organization.
   3. Choose **Save**.

   > **Note**
   >
   > It may take up to 24 hours to update the configuration for the member accounts.

Using the Accounts page

1. In the navigation pane, choose **Accounts**.
2. On the **Accounts** page, choose **Auto-enable** preferences before **Add accounts by invitation**.
3. In the **Manage auto-enable preferences** window, choose **Enable for all accounts** under **RDS Login Activity Monitoring**.
4. Choose **Save**.

If you can't use the **Enable for all accounts** option, see Selectively enable RDS Protection for member accounts.

API/CLI

To selectively enable or disable RDS Protection for your member accounts, invoke the updateMemberDetectors API operation using your own *detector ID*.

Alternatively, you can use AWS CLI to enable RDS Protection. Run the following command, and replace *12abc34d567e8fa901bc2d34e56789f0* with your account's detector ID and *us-east-1* with the Region where you want to enable RDS Protection.

To find the `detectorId` for your account and current Region, see the **Settings** page in the https://console.aws.amazon.com/guardduty/ console, or run the ListDetectors API.

```
aws guardduty update-member-detectors --detector-id 12abc34d567e8fa901bc2d34e56789f0
 --region us-east-1 --account-ids 111122223333 --features '[{"name":
 "RDS_LOGIN_EVENTS", "status": "ENABLED"}]'
```

You can also pass a list of account IDs separated by a space.

When the code has successfully executed, it returns an empty list of `UnprocessedAccounts`. If there were any problems changing the detector settings for an account, that account ID is listed along with a summary of the issue.

Enable RDS Protection for all existing active member accounts

Choose your preferred access method to enable RDS Protection for all the existing active member accounts in your organization. The member accounts that already have GuardDuty enabled, are referred to as existing active members.

Console

1. Sign in to the AWS Management Console and open the GuardDuty console at https://console.aws.amazon.com/guardduty/.

   Sign in using the delegated GuardDuty administrator account credentials.
2. In the navigation pane, choose **RDS Protection**.
3. On the **RDS Protection** page, you can view the current status of the configuration. Under the **Active member accounts** section, choose **Actions**.
4. From the **Actions** dropdown menu, choose **Enable for all existing active member accounts**.
5. Choose **Confirm**.

API/CLI

Run the updateMemberDetectors API operation using your own *detector ID*.

Alternatively, you can use AWS CLI to enable RDS Protection. Run the following command, and replace *12abc34d567e8fa901bc2d34e56789f0* with your account's detector ID and *us-east-1* with the Region where you want to enable RDS Protection.

To find the `detectorId` for your account and current Region, see the **Settings** page in the https://console.aws.amazon.com/guardduty/ console, or run the ListDetectors API.

```
aws guardduty update-member-detectors --detector-id 12abc34d567e8fa901bc2d34e56789f0
 --region us-east-1 --account-ids 111122223333 --features '[{"name":
 "RDS_LOGIN_EVENTS", "status": "ENABLED"}]'
```

You can also pass a list of account IDs separated by a space.

When the code has successfully executed, it returns an empty list of `UnprocessedAccounts`. If there were any problems changing the detector settings for an account, that account ID is listed along with a summary of the issue.

Auto-enable RDS Protection for new member accounts

Choose your preferred access method to enable RDS login activity for new accounts that join your organization.

Console

The delegated GuardDuty administrator account can enable for new member accounts in an organization through the console, using either the **RDS Protection** or **Accounts** page.

To auto-enable RDS Protection for new member accounts

1. Open the GuardDuty console at https://console.aws.amazon.com/guardduty/.

   Make sure to use the delegated GuardDuty administrator account credentials.
2. Do one of the following:
   - Using the **RDS Protection** page:
     1. In the navigation pane, choose **RDS Protection**.
     2. On the **RDS Protection** page, choose **Edit**.
     3. Choose **Configure accounts manually**.
     4. Select **Automatically enable for new member accounts**. This step ensures that whenever a new account joins your organization, RDS Protection will be automatically enabled for their account. Only the organization delegated GuardDuty administrator account can modify this configuration.
     5. Choose **Save**.
   - Using the **Accounts** page:
     1. In the navigation pane, choose **Accounts**.
     2. On the **Accounts** page, choose **Auto-enable** preferences.
     3. In the **Manage auto-enable preferences** window, select **Enable for new accounts** under **RDS Login Activity Monitoring**.
     4. Choose **Save**.

API/CLI

To selectively enable or disable RDS Protection for your member accounts, invoke the UpdateOrganizationConfiguration API operation using your own *detector ID*.

Alternatively, you can use AWS CLI to enable RDS Protection. Run the following command, and replace *12abc34d567e8fa901bc2d34e56789f0* with your account's detector ID and *us-east-1* with the Region where you want to enable RDS Protection. If you don't want to enable it for all the new accounts joining the organization, set `autoEnable` to `NONE`.

To find the `detectorId` for your account and current Region, see the **Settings** page in the https://console.aws.amazon.com/guardduty/ console, or run the ListDetectors API.

```
aws guardduty update-organization-configuration --detector-
id 12abc34d567e8fa901bc2d34e56789f0 --region us-east-1 --auto-enable --features
 '[{"Name": "RDS_LOGIN_EVENTS", "AutoEnable": "NEW"}]'
```

When the code has successfully executed, it returns an empty list of `UnprocessedAccounts`. If there were any problems changing the detector settings for an account, that account ID is listed along with a summary of the issue.

Selectively enable RDS Protection for member accounts

Choose your preferred access method to selectively enable monitoring RDS login activity for member accounts.

Console

1. Open the GuardDuty console at https://console.aws.amazon.com/guardduty/.

   Make sure to use the delegated GuardDuty administrator account credentials.
2. In the navigation pane, choose **Accounts**.

   On the **Accounts** page, review the **RDS login activity** column for the status of your member account.
3. **To selectively enable or disable RDS login activity**

   Select the account for which you want to configure RDS Protection. You can select multiple accounts at a time. In the **Edit Protection Plans** dropdown menu, choose **RDS Login Activity**, and then choose the appropriate option.

API/CLI

To selectively enable or disable RDS Protection for your member accounts, invoke the updateMemberDetectors API operation using your own *detector ID*.

Alternatively, you can use AWS CLI to enable RDS Protection. Run the following command, and replace *12abc34d567e8fa901bc2d34e56789f0* with your account's detector ID and *us-east-1* with the Region where you want to enable RDS Protection.

To find the `detectorId` for your account and current Region, see the **Settings** page in the https://console.aws.amazon.com/guardduty/ console, or run the ListDetectors API.

```
aws guardduty update-member-detectors --detector-id 12abc34d567e8fa901bc2d34e56789f0
 --region us-east-1 --account-ids 111122223333 --features '[{"Name":
 "RDS_LOGIN_EVENTS", "Status": "ENABLED"}]'
```

> **Note**
>
> You can also pass a list of account IDs separated by a space.

When the code has successfully executed, it returns an empty list of `UnprocessedAccounts`. If there were any problems changing the detector settings for an account, that account ID is listed along with a summary of the issue.

Enabling RDS Protection for a standalone account

A standalone account owns the decision to enable or disable a protection plan in their AWS account in a specific AWS Region.

If your account is associated with a GuardDuty administrator account through AWS Organizations, or by the method of invitation, this section doesn't apply to your account. For more information, see Enabling RDS Protection in multiple-account environments.

After you enable RDS Protection, GuardDuty will start monitoring RDS login activity for the supported databases in your account.

Choose your preferred access method to configure RDS Protection for a standalone account.

Console

1. Open the GuardDuty console at https://console.aws.amazon.com/guardduty/.
2. In the navigation pane, choose **RDS Protection**.
3. The **RDS Protection** page shows the current status for your account. Choose **Enable** to enable RDS Protection.
4. Choose **Confirm** to save your selection.

API/CLI

Run the updateDetector API operation using your own regional detector ID and passing the `features` object name as `RDS_LOGIN_EVENTS` and `status` as `ENABLED`.

Alternatively, you can use AWS CLI to enable RDS Protection. Run the following command, and replace *12abc34d567e8fa901bc2d34e56789f0* with your account's detector ID and *us-east-1* with the Region where you want to enable RDS Protection.

To find the `detectorId` for your account and current Region, see the **Settings** page in the https://console.aws.amazon.com/guardduty/ console, or run the ListDetectors API.

```
aws guardduty update-detector --detector-id 12abc34d567e8fa901bc2d34e56789f0 --
region us-east-1 --features '[{"Name" : "RDS_LOGIN_EVENTS", "Status" : "ENABLED"}]'
```

GuardDuty Lambda Protection

Lambda Protection helps you identify potential security threats when an AWS Lambda function gets invoked in your AWS environment. When you enable Lambda Protection, GuardDuty starts monitoring Lambda network activity logs. This includes VPC Flow Logs from all Lambda functions for your account (including those logs that don't use VPC networking) and logs that get generated when Lambda function gets invoked. When GuardDuty identifies suspicious network traffic that is indicative of the presence of a potentially malicious piece of code in your Lambda function, GuardDuty generates one or more Lambda Protection finding types.

30-day free trial

The following list explains how the 30-day free trial works for your account:

- When you enable GuardDuty in an AWS account in a new Region for the first time, you get a 30-day free trial. In this case, GuardDuty will also enable Lambda Protection, which is included in the free trial.
- When you are already using GuardDuty and decide to enable Lambda Protection for the first time, your account in this Region will get a 30-day free trial for Lambda Protection.
- You can choose to disable Lambda Protection in any Region at any time.
- During the 30-day free trial, you can get an estimate of your usage costs in that account and Region. After the 30-day free trial ends, Lambda Protection doesn't get disabled automatically. Your account in this Region will start incurring usage cost. For more information, see Estimating GuardDuty usage cost.

Lambda network activity logs are subject to change, including expansion to other network activity such as DNS query data generated by invoking the Lambda functions. The expansion into other forms of network activity monitoring will increase the volume of data that GuardDuty will process for Lambda Protection. This will directly impact the usage cost of Lambda Protection. Whenever GuardDuty starts monitoring an additional network activity log, it will provide a notice to the accounts that have turned on Lambda Protection, at least 30 days prior to the release.

> **Note**
>
> Lambda Network Activity Monitoring doesn't include the logs for Lambda@Edge functions.

Lambda Network Activity Monitoring

When you enable Lambda Protection, GuardDuty monitors Lambda network activity logs that gets generated when a Lambda function, associated to your account, gets invoked. This helps you detect potential security threats to the Lambda function. For Lambda functions that are configured to use VPC networking, you don't need to enable VPC flow logs for the elastic network interfaces (ENI) created by Lambda for GuardDuty. GuardDuty only charges for the amount of Lambda network activity logs data processed (in GB) to generate a finding. GuardDuty optimizes cost by applying smart filters and analyzing a subset of Lambda network activity logs that are relevant to threat detection.

GuardDuty doesn't manage your Lambda network activity logs (including VPC and non-VPC flow logs), or make them accessible in your account.

Enabling Lambda Protection in multiple-account environments

In a multi-account environment, only the delegated GuardDuty administrator account has the option to enable or disable Lambda Protection for the member accounts in their organization. The GuardDuty member accounts can't modify this configuration from their accounts. The delegated GuardDuty administrator account manages member accounts using AWS Organizations. The delegated GuardDuty administrator account can choose to auto-enable Lambda Network Activity Monitoring for all the new accounts as they join the organization. For more information about multiple-account environments, see Managing multiple accounts in Amazon GuardDuty.

Enabling Lambda Protection for delegated GuardDuty administrator account

Choose your preferred access method to enable or disable Lambda Network Activity Monitoring for delegated GuardDuty administrator account.

Console

1. Open the GuardDuty console at https://console.aws.amazon.com/guardduty/.
2. In the navigation pane, under **Settings**, choose **Lambda Protection**.
3. On the **Lambda Protection** page, choose **Edit**.
4. Do one of the following:

Using Enable for all accounts

- Choose **Enable for all accounts**. This will enable the protection plan for all the active GuardDuty accounts in your AWS organization, including the new accounts that join the organization.
- Choose **Save**.

Using Configure accounts manually

- To enable the protection plan only for the delegated GuardDuty administrator account account, choose **Configure accounts manually**.
- Choose **Enable** under the **delegated GuardDuty administrator account (this account)** section.
- Choose **Save**.

API/CLI

Run the [updateDetector](https://docs.aws.amazon.com/guardduty/latest/APIReference/API_UpdateDetector.html) API operation using your own regional detector ID and passing the `features` object name as `LAMBDA_NETWORK_LOGS` and `status` as `ENABLED`.

Alternatively, you can use AWS CLI to enable Lambda Protection. Run the following command, and replace *`12abc34d567e8fa901bc2d34e56789f0`* with your account's detector ID and *`us-east-1`* with the Region where you want to enable Lambda Protection.

To find the `detectorId` for your account and current Region, see the **Settings** page in the https://console.aws.amazon.com/guardduty/ console, or run the ListDetectors API.

```
aws guardduty update-detector --detector-id 12abc34d567e8fa901bc2d34e56789f0 --
region us-east-1 --features '[{"Name": "LAMBDA_NETWORK_LOGS", "Status": "ENABLED"}]'
```

Auto-enable Lambda Network Activity Monitoring for all member accounts

Choose your preferred access method to enable the Lambda Network Activity Monitoring feature for all member accounts. This includes existing member accounts and the new accounts that join the organization.

Console

1. Sign in to the AWS Management Console and open the GuardDuty console at https://console.aws.amazon.com/guardduty/.

   Make sure to use the delegated GuardDuty administrator account credentials.

2. Do one of the following:

   Using the Lambda Protection page

   1. In the navigation pane, choose **Lambda Protection**.
   2. Choose **Enable for all accounts**. This action automatically enables Lambda Network Activity Monitoring for both existing and new accounts in the organization.
   3. Choose **Save**.

   > **Note**
   >
   > It may take up to 24 hours to update the configuration for the member accounts.

   Using the Accounts page

   1. In the navigation pane, choose **Accounts**.
   2. On the **Accounts** page, choose **Auto-enable** preferences before **Add accounts by invitation**.
   3. In the **Manage auto-enable preferences** window, choose **Enable for all accounts** under **Lambda Network Activity Monitoring**.

   > **Note**
   >
   > By default, this action automatically turns on the **Auto-enable GuardDuty for new member accounts** option.

   4. Choose **Save**.

   If you can't use the **Enable for all accounts** option, see Selectively enable or disable Lambda Network Activity Monitoring for member accounts.

API/CLI

To selectively enable or disable Lambda Network Activity Monitoring for your member accounts, invoke the updateMemberDetectors API operation using your own *detector ID*.

Alternatively, you can use AWS CLI to enable Lambda Protection. Run the following command, and replace *12abc34d567e8fa901bc2d34e56789f0* with your account's detector ID and *us-east-1* with the Region where you want to enable Lambda Protection.

To find the `detectorId` for your account and current Region, see the **Settings** page in the https://console.aws.amazon.com/guardduty/ console, or run the ListDetectors API.

```
aws guardduty update-member-detectors --detector-id 12abc34d567e8fa901bc2d34e56789f0
 --account-ids 111122223333 --region us-east-1--features '[{"Name":
 "LAMBDA_NETWORK_LOGS", "Status": "ENABLED"}]'
```

You can also pass a list of account IDs separated by a space.

When the code has successfully executed, it returns an empty list of `UnprocessedAccounts`. If there were any problems changing the detector settings for an account, that account ID is listed along with a summary of the issue.

Enable Lambda Network Activity Monitoring for all existing active member accounts

Choose your preferred access method to enable Lambda Network Activity Monitoring for all the existing active member accounts in the organization.

Console

To configure Lambda Network Activity Monitoring for all existing active member accounts

1. Sign in to the AWS Management Console and open the GuardDuty console at https://console.aws.amazon.com/guardduty/.

   Sign in using the delegated GuardDuty administrator account credentials.
2. In the navigation pane, choose **Lambda Protection**.
3. On the **Lambda Protection** page, you can view the current status of the configuration. Under the **Active member accounts** section, choose **Actions**.

4. From the **Actions** dropdown menu, choose **Enable for all existing active member accounts**.
5. Choose **Confirm**.

API/CLI

To selectively enable or disable Lambda Network Activity Monitoring for your member accounts, invoke the [updateMemberDetectors](#) API operation using your own *detector ID*.

Alternatively, you can use AWS CLI to enable Lambda Protection. Run the following command, and replace *12abc34d567e8fa901bc2d34e56789f0* with your account's detector ID and *us-east-1* with the Region where you want to enable Lambda Protection.

```
aws guardduty update-member-detectors --detector-id 12abc34d567e8fa901bc2d34e56789f0
 --region us-east-1 --account-ids 111122223333 --features '[{"Name":
 "LAMBDA_NETWORK_LOGS", "Status": "ENABLED"}]'
```

You can also pass a list of account IDs separated by a space.

When the code has successfully executed, it returns an empty list of `UnprocessedAccounts`. If there were any problems changing the detector settings for an account, that account ID is listed along with a summary of the issue.

Auto-enable Lambda Network Activity Monitoring for new member accounts

Choose your preferred access method to enable Lambda Network Activity Monitoring for new accounts that join your organization.

Console

The delegated GuardDuty administrator account can enable Lambda Network Activity Monitoring for new member accounts in an organization, using either the **Lambda Protection** or **Accounts** page.

To auto-enable Lambda Network Activity Monitoring for new member accounts

1. Open the GuardDuty console at https://console.aws.amazon.com/guardduty/.

   Make sure to use the delegated GuardDuty administrator account credentials.

2. Do one of the following:

   - Using the **Lambda Protection** page:

     1. In the navigation pane, choose **Lambda Protection**.
     2. On the **Lambda Protection** page, choose **Edit**.
     3. Choose **Configure accounts manually**.
     4. Select **Automatically enable for new member accounts**. This step ensures that whenever a new account joins your organization, Lambda Protection will be automatically enabled for their account. Only the organization delegated GuardDuty administrator account can modify this configuration.
     5. Choose **Save**.

   - Using the **Accounts** page:

     1. In the navigation pane, choose **Accounts**.
     2. On the **Accounts** page, choose **Auto-enable** preferences.
     3. In the **Manage auto-enable preferences** window, select **Enable for new accounts** under **Lambda Network Activity Monitoring**.
     4. Choose **Save**.

API/CLI

To enable Lambda Network Activity Monitoring for new member accounts, invoke the UpdateOrganizationConfiguration API operation using your own *detector ID*.

Alternatively, you can use AWS CLI to enable Lambda Protection. The following example shows how you can enable Lambda Network Activity Monitoring for a single member account. Replace *12abc34d567e8fa901bc2d34e56789f0* with your account's detector ID and *us-east-1* with the Region where you want to enable Lambda Protection. If you don't want to enable it for all the new accounts joining the organization, set `AutoEnable` to `NONE`.

To find the `detectorId` for your account and current Region, see the **Settings** page in the https://console.aws.amazon.com/guardduty/ console, or run the ListDetectors API.

```
aws guardduty update-organization-configuration --detector-
id 12abc34d567e8fa901bc2d34e56789f0 --region us-east-1 --auto-enable --features
 '[{"Name": "LAMBDA_NETWORK_LOGS", "AutoEnable": "NEW"}]'
```

When the code has successfully executed, it returns an empty list of `UnprocessedAccounts`. If there were any problems changing the detector settings for an account, that account ID is listed along with a summary of the issue.

Selectively enable or disable Lambda Network Activity Monitoring for member accounts

Choose your preferred access method to selectively enable or disable Lambda Network Activity Monitoring for member accounts.

Console

1. Open the GuardDuty console at https://console.aws.amazon.com/guardduty/.

   Make sure to use the delegated GuardDuty administrator account credentials.
2. In the navigation pane, under **Settings**, choose **Accounts**.

   On the **Accounts** page, review the **Lambda Network Activity Monitoring** column. It indicates whether or not Lambda Network Activity Monitoring is enabled.
3. Choose the account for which you want to configure Lambda Protection. You can choose multiple accounts at a time.
4. From the **Edit Protection Plans** dropdown menu, choose **Lambda Network Activity Monitoring**, and then choose an appropriate action.

API/CLI

Invoke the updateMemberDetectors API using your own *`detector ID`*.

Alternatively, you can use AWS CLI to enable Lambda Protection. Replace *`12abc34d567e8fa901bc2d34e56789f0`* with your account's detector ID and *`us-east-1`* with the Region where you want to enable Lambda Protection.

To find the `detectorId` for your account and current Region, see the **Settings** page in the https://console.aws.amazon.com/guardduty/ console, or run the ListDetectors API.

```
aws guardduty update-member-detectors --detector-id 12abc34d567e8fa901bc2d34e56789f0
 --region us-east-1 --account-ids 111122223333 --features '[{"Name":
 "LAMBDA_NETWORK_LOGS", "Status": "ENABLED"}]'
```

You can also pass a list of account IDs separated by a space.

When the code has successfully executed, it returns an empty list of `UnprocessedAccounts`. If there were any problems changing the detector settings for an account, that account ID is listed along with a summary of the issue.

Enabling Lambda Protection for a standalone account

A standalone account owns the decision to enable or disable a protection plan in their AWS account in a specific AWS Region.

If your account is associated with a GuardDuty administrator account through AWS Organizations, or by the method of invitation, this section doesn't apply to your account. For more information, see Enabling Lambda Protection in multiple-account environments.

After you enable Lambda Protection, GuardDuty will start monitoring Lambda Network Activity Monitoring in your account.

Choose your preferred access method to configure Lambda Protection for a standalone account.

Console

1. Open the GuardDuty console at https://console.aws.amazon.com/guardduty/.
2. In the navigation pane, under **Settings**, choose **Lambda Protection**.
3. The Lambda Protection page shows the current status for your account. Choose **Enable** to enable Lambda Protection in your account.
4. Choose **Confirm** to save your selection.

API/CLI

Run the updateDetector API operation using your own regional detector ID and passing the `features` object name as `LAMBDA_NETWORK_LOGS` and `status` as `ENABLED`.

Alternatively, you can use AWS CLI to enable Lambda Protection. Run the following command, and replace *`12abc34d567e8fa901bc2d34e56789f0`* with your account's detector ID and *`us-east-1`* with the Region where you want to enable Lambda Protection.

To find the `detectorId` for your account and current Region, see the **Settings** page in the https://console.aws.amazon.com/guardduty/ console, or run the ListDetectors API.

```
aws guardduty update-detector --detector-id 12abc34d567e8fa901bc2d34e56789f0
 --region us-east-1 --features [{"Name" : "LAMBDA_NETWORK_LOGS", "Status" :
 "ENABLED"}]'
```

Protecting AI workloads with GuardDuty

Amazon GuardDuty foundational threat detection and Lambda Protection helps you to better secure and detect threats to AI workloads built on AWS.

The foundational GuardDuty threat detection monitors AWS CloudTrail management events to detect suspicious and malicious activity in generative AI workloads created by using AWS services, including Amazon Bedrock and Amazon SageMaker AI. For example, GuardDuty can identify activities such as:

- Unusual removal of Amazon Bedrock security guardrails
- Change of model training data source that can potentially lead to data poisoning attack
- Suspicious Amazon Bedrock model invocation
- Unusual notebook instance or training job creation in SageMaker AI
- Exfiltrated Amazon Elastic Compute Cloud credentials that may have been used to call APIs in Amazon Bedrock, Amazon SageMaker AI, or self-managed AI workloads on EC2 instances, EKS clusters, or ECS tasks.

GuardDuty Lambda Protection can help detect potential threats related Amazon Bedrock agents. This may include suspicious network activity such as cryptomining, and communicating with malicious command and control servers that can be caused by supply chain attack or complex prompting.

The following video shows how the associated findings would look.

The following video shows how the associated findings would look. Using Amazon GuardDuty to monitor and secure your AI workloads built on AWS

Multiple accounts in Amazon GuardDuty

When your AWS environment has multiple accounts, you can manage them by designating one AWS account as the administrator account. You can then associate the multiple AWS accounts with this administrator account as its member accounts. With this configuration, a designated GuardDuty administrator account can assess and monitor the overall security of your organization. The administrator account can also perform account management tasks, such as reviewing all generated findings and configuring protection plans within GuardDuty.

In GuardDuty, an organization consists of a delegated GuardDuty administrator account and one or more associated member accounts. You can associate the accounts in two ways – by integrating with AWS Organizations, or by using a legacy method of sending and accepting membership invitations in the GuardDuty console. GuardDuty recommends that you integrate with AWS Organizations.

AWS Organizations is a global account management service that enables AWS administrators to consolidate and centrally manage multiple AWS accounts. It provides account management and consolidated billing features that are designed to support budgetary, security, and compliance needs. It's offered at no additional charge and it integrates with multiple AWS services, including Macie, AWS Security Hub, and Amazon GuardDuty. For more information, see the AWS Organizations User Guide.

Contents

- Understanding the relationship between GuardDuty administrator account and member accounts
- Managing GuardDuty accounts with AWS Organizations
- Managing GuardDuty accounts by invitation
- GuardDuty considerations for exporting member account details in CSV format

Understanding the relationship between GuardDuty administrator account and member accounts

When you use GuardDuty in a multiple-account environment, the administrator account can manage certain aspects of GuardDuty on behalf of the member accounts. An administrator account can perform the following primary functions:

- **Add and remove associated member accounts –** The process by which an administrator account can do this differs based on how you manage the accounts – through AWS Organizations or by GuardDuty invitation method.

  GuardDuty recommends managing your member accounts through AWS Organizations.
- **Delegated GuardDuty administrator account enabling GuardDuty in management account –** If the AWS Organizations management account ever disables GuardDuty, the delegated GuardDuty administrator account can enable GuardDuty in the management account. However, it is required that the management account must have not explicitly deleted the [Service-linked role permissions for GuardDuty](#).
- *Configure status of member accounts* **–** An administrator account can enable or disable the status of GuardDuty protection plans, and enable, suspend, or disable the status of GuardDuty on behalf of associated member accounts.

  Delegated GuardDuty administrator account managed with AWS Organizations can automatically enable GuardDuty when the AWS accounts are added as members.
- **Customize when to generate findings –** An administrator account can customize findings within the GuardDuty network by creating and managing suppression rules, trusted IP lists, and threat lists. In a multiple-account environment, support to configure these features is available only to an delegated GuardDuty administrator account. A member account can't update this configuration.

The following table details the relationship between GuardDuty administrator account and member accounts.

Key for the table

- **Self –** An account can perform the listed action only for their own account.
- **Any –** An account can perform the listed action for any associated account.
- **All –** An account can perform the listed action and it applies to all the associated accounts. Usually, the account taking this action is a designated GuardDuty administrator account
- **Cells with dash (–) –** Table cells with dash (–) indicate that the account can't perform the listed action.

Action	Through AWS Organizations	By invitation

	Delegated GuardDuty administrator account	Associated member account	GuardDuty administrator account	Associated member account
Enable GuardDuty	Any	–	Self	Self
Enable GuardDuty automatically for the entire organization (`ALL`, `NEW`, `NONE`)	All	–	–	–
View all Organizations member accounts regardless of GuardDuty status	Any	–	–	–
Generate sample findings	Self	Self	Self	Self
View all GuardDuty findings	Any	Self	Any	Self
Archive GuardDuty findings	Any	–	Any	–
Apply suppression rules	All	–	All	–

Create trusted IP list or threat lists	All	–	All	–
Update trusted IP list or threat lists	All	–	All	–
Delete trusted IP list or threat lists	All	–	All	–
Set EventBridge notification frequency	All	–	All	–
Set Amazon S3 location for exporting findings	All	Self	All	Self
Enable one or more optional protection plans for the entire organization (`ALL`, `NEW`, `NONE`) This doesn't include Malware Protection for S3.	All	–	–	–

Enable any GuardDuty protection plan for individual accounts This doesn't include Malware Protection for EC2 and Malware Protection for S3.	Any	–	Any	–
Malware Protection for EC2	Any	–	Self	–
Malware Protection for EC2 – On-demand malware scan	Any	Self	Self	Self
Malware Protection for S3	–	Self	–	Self
Disassociate a member account	Any[+]	–	Any	–
Disassociate from an administrator account	–	–	–	Self

Delete a disassociated member account	Any	–	Any	–
Suspend GuardDuty	Any*	–	Any*	–
Disable GuardDuty	Any*	–	Any*	–

+Indicates that the delegated GuardDuty administrator account can take this action only if they have not set up the auto-enable preferences to `ALL` the organization members.

*Indicates that a delegated GuardDuty administrator account can't disable GuardDuty in a member account directly. The delegated GuardDuty administrator account must first disassociate the member account, and then delete them. After this, each member account can disable GuardDuty in their own accounts. For more information about performing these tasks in your organization, see Continually managing your member accounts within GuardDuty.

Managing GuardDuty accounts with AWS Organizations

In an AWS organization, the management account can designate any account within this organization as the delegated GuardDuty administrator account. For this administrator account, GuardDuty gets enabled automatically only in the current AWS Region. By default, the administrator account can enable and manage GuardDuty for all the member accounts in the organization within that Region. The administrator account can view and add members to this AWS organization.

The following sections will walk you through various tasks that you may perform as a delegated GuardDuty administrator account.

Contents

- Considerations and recommendations for using GuardDuty with AWS Organizations
- Permissions required to designate a delegated GuardDuty administrator account
- Designating a delegated GuardDuty administrator account
- Setting organization auto-enable preferences
- Adding members to the organization

- (Optional) Enable protection plans for existing member accounts
- Continually managing your member accounts within GuardDuty
- Suspending GuardDuty for member account
- Disassociating (removing) member account from administrator account
- Deleting member accounts from GuardDuty organization
- Changing the delegated GuardDuty administrator account

Considerations and recommendations for using GuardDuty with AWS Organizations

The following considerations and recommendations can help you understand how a delegated GuardDuty administrator account operates in GuardDuty:

A delegated GuardDuty administrator account can manage a maximum of 50,000 members.

There is a limit of 50,000 member accounts per delegated GuardDuty administrator account. This includes member accounts that are added through AWS Organizations or those who accepted the GuardDuty administrator account's invitation to join their organization. However, there could be more than 50,000 accounts in your AWS organization.

If you exceed the 50,000 member accounts limit, you will receive a notification from CloudWatch, AWS Health Dashboard, and an email to the designated delegated GuardDuty administrator account.

A delegated GuardDuty administrator account is Regional.

Unlike AWS Organizations, GuardDuty is a Regional service. The delegated GuardDuty administrator accounts and their member accounts must be added through AWS Organizations in each desired Region where you have GuardDuty enabled. If the organization management account designates a delegated GuardDuty administrator account in only US East (N. Virginia), then delegated GuardDuty administrator account will only manage member accounts added to the organization in that Region. For more information about feature parity in Regions where GuardDuty is available, see Regions and endpoints.

Special cases for opt-in Regions

- When a delegated GuardDuty administrator account opts out of an opt-in Region, even if your organization has the GuardDuty auto-enable configuration set to either new member accounts only (`NEW`) or all member accounts (`ALL`), GuardDuty cannot be enabled for any member account in the organization that currently has GuardDuty disabled. For information about the configuration of your member accounts, open **Accounts** in the GuardDuty console navigation pane or use the ListMembers API.

- When working with the GuardDuty auto-enable configuration set to `NEW`, ensure that the following sequence is met:

  1. The member accounts opt-in to an opt-in Region.
  2. Add the member accounts to your organization in AWS Organizations.

  If you change the order of these steps, the GuardDuty auto-enable setting with `NEW` **will not** work in the specific opt-in Region because the member account is no longer new to the organization. GuardDuty provides two alternate solutions:

  - Set the GuardDuty auto-enable configuration to `ALL`, that includes new and existing members accounts. In this case, the order of these steps is not relevant.
  - If a member account is already a part of your organization, manage the GuardDuty configuration for this account individually in the specific opt-in Region by using the GuardDuty console or the API.

Required for an AWS organization to have the same delegated GuardDuty administrator account across all the AWS Regions.

You must designate one member account as the delegated GuardDuty administrator account across all the AWS Regions where GuardDuty is enabled. For example, if you designate a member account *`111122223333`* in *`Europe (Ireland)`*, you can't designate another member account *`555555555555`* in *`Canada (Central)`*. It is required that you use the same account as delegated GuardDuty administrator account in all other Regions.

You can designate a new delegated GuardDuty administrator account at any point in time. For more information about removing the existing delegated GuardDuty administrator account, see Changing the delegated GuardDuty administrator account.

Not recommended to set your organization's management account as the delegated GuardDuty administrator account.

Your organization's management account can be the delegated GuardDuty administrator account. However, the AWS security best practices follow the principle of least privilege and doesn't recommend this configuration.

Changing a delegated GuardDuty administrator account does not disable GuardDuty for member accounts.

If you remove a delegated GuardDuty administrator account, GuardDuty removes all the member accounts associated with this delegated GuardDuty administrator account. GuardDuty still remains enabled for all these member accounts.

Permissions required to designate a delegated GuardDuty administrator account

To start using Amazon GuardDuty with AWS Organizations, the AWS Organizations management account for the organization designates an account as the delegated GuardDuty administrator account. This enables GuardDuty as a trusted service in AWS Organizations. It also enables GuardDuty for the delegated GuardDuty administrator account and also allows the delegated administrator account to enable and manage GuardDuty for other accounts in the organization in the current Region. For information about how these permissions are granted, see Using AWS Organizations with other AWS services.

As the AWS Organizations management account, before you designate the delegated GuardDuty administrator account for your organization, verify that you can perform the following GuardDuty action: `guardduty:EnableOrganizationAdminAccount`. This action allows you to designate the delegated GuardDuty administrator account for your organization by using GuardDuty. You must also ensure that you are allowed to perform the AWS Organizations actions that help you retrieve information about your organization.

To grant these permissions, include the following statement in an AWS Identity and Access Management (IAM) policy for your account:

```
{
    "Sid": "PermissionsForGuardDutyAdmin",
    "Effect": "Allow",
    "Action": [
        "guardduty:EnableOrganizationAdminAccount",
        "organizations:EnableAWSServiceAccess",
        "organizations:RegisterDelegatedAdministrator",
        "organizations:ListDelegatedAdministrators",
        "organizations:ListAWSServiceAccessForOrganization",
        "organizations:DescribeOrganizationalUnit",
        "organizations:DescribeAccount",
```

```
        "organizations:DescribeOrganization",
        "organizations:ListAccounts"
    ],
    "Resource": "*"
}
```

If you want to designate your AWS Organizations management account as the delegated GuardDuty administrator account, your account will also need the IAM action: `CreateServiceLinkedRole`. This action allows you to initialize GuardDuty for the management account. However, review Considerations and recommendations for using GuardDuty with AWS Organizations before you proceed to add the permissions.

To continue with designating the management account as the delegated GuardDuty administrator account, add the following statement to the IAM policy and replace *111122223333* with the AWS account ID of your organization's management account:

```
{
 "Sid": "PermissionsToEnableGuardDuty"
 "Effect": "Allow",
 "Action": [
  "iam:CreateServiceLinkedRole"
 ],
 "Resource": "arn:aws:iam::111122223333:role/aws-service-role/guardduty.amazonaws.com/
AWSServiceRoleForAmazonGuardDuty",
 "Condition": {
  "StringLike": {
   "iam:AWSServiceName": "guardduty.amazonaws.com"
  }
 }
}
```

Designating a delegated GuardDuty administrator account

This section provides steps to designate a delegated administrator in the GuardDuty organization.

As a management account of the AWS organization, make sure that you read through the Considerations and recommendations on how a delegated GuardDuty administrator account operates. Before proceeding, ensure that you have Permissions required to designate a delegated GuardDuty administrator account.

Choose a preferred access method to designate a delegated GuardDuty administrator account for your organization. Only a management account can perform this step.

Console

1. Open the GuardDuty console at https://console.aws.amazon.com/guardduty/.

   To sign in, use the management account credentials for your AWS Organizations organization.

2. By using the AWS Region selector in the upper-right corner of the page, select the Region in which you want to designate the delegated GuardDuty administrator account for your organization.

3. Do one of the following, depending on whether GuardDuty is enabled for your management account in the current Region:

   - If GuardDuty is not enabled, select **Amazon GuardDuty - all features** and choose **Get started**. This action will take you to the **Welcome to GuardDuty** page.
   - If GuardDuty is enabled, choose **Settings** in the navigation pane.

4. Under **Delegated administrator**, enter the 12-digit AWS account ID of the account that you want to designate as the delegated GuardDuty administrator account for the organization.

   Make sure to enable GuardDuty for your newly designated delegated GuardDuty administrator account, otherwise it won't be able to take any action.

5. Choose **Delegate**.

6. (Recommended) Repeat the preceding steps to designate the delegated GuardDuty administrator account in each AWS Region where you have GuardDuty enabled.

API/CLI

1. Run enableOrganizationAdminAccount using the credentials of the AWS account of the organization's management account.

   - Alternatively, you can use AWS Command Line Interface to do this. The following AWS CLI command designates a delegated GuardDuty administrator account for your current Region only. Run the following AWS CLI command and make sure to replace *111111111111* with the AWS account ID of the account you want to designate as a delegated GuardDuty administrator account:

```
aws guardduty enable-organization-admin-account --admin-account-
id 111111111111
```

To designate the delegated GuardDuty administrator account for other Regions, specify the Region in the AWS CLI command. The following example demonstrates how to enable a delegated GuardDuty administrator account in US West (Oregon). Make sure to replace *us-west-2* with the Region for which you want to assign the delegated GuardDuty administrator account.

```
aws guardduty enable-organization-admin-account --admin-account-
id 111111111111 --region us-west-2
```

For information about the AWS Regions where GuardDuty is available, see Regions and endpoints.

If GuardDuty is disabled for your delegated GuardDuty administrator account, it won't be able to take any action. If not already done so, make sure to enable GuardDuty for the newly designated delegated GuardDuty administrator account.

2. (Recommended) repeat the preceding steps to designate the delegated GuardDuty administrator account in each AWS Region where you have GuardDuty enabled.

Setting organization auto-enable preferences

The auto-enable organization feature in GuardDuty helps you set the same GuardDuty and protection plans status for `ALL` existing or `NEW` member accounts in your organization, in a single step. Similarly, you can also specify when you don't want to take any action on the member accounts, by choosing `NONE`. The following steps explain these settings and also indicate when you would want to use a specific setting.

> **Note**
>
> You can set auto-enable preferences for all the protection plans except Malware Protection for S3.

Choose a preferred access method to update the auto-enable preferences for the organization.

Console

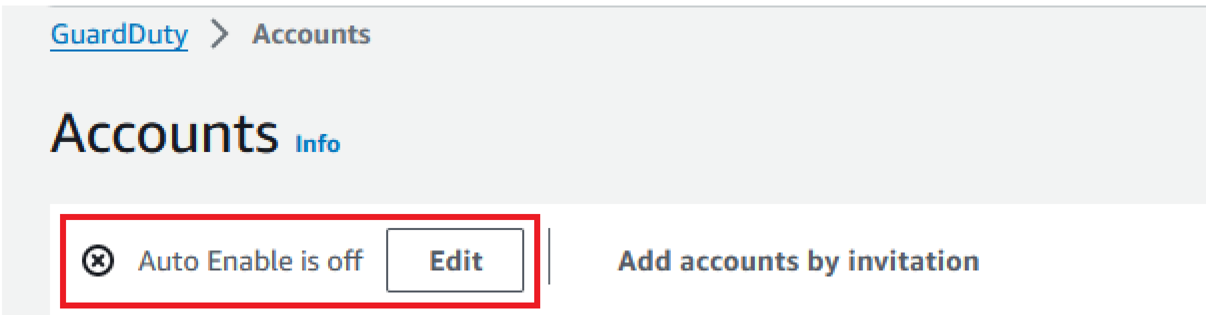
1. Open the GuardDuty console at https://console.aws.amazon.com/guardduty/.

   To sign in, use the GuardDuty administrator account credentials.

2. In the navigation pane, choose **Accounts**.

   The **Accounts** page provides configuration options to the GuardDuty administrator account to **Auto-enable** GuardDuty and the optional protection plans on behalf of the member accounts that belong to the organization.

3. To update the existing auto-enable settings, choose **Edit**.



   This support is available to configure GuardDuty and all of the supported optional protection plans in your AWS Region. You can select one of the following configuration options for GuardDuty on behalf of your member accounts:

   - **Enable for all accounts (ALL)** – Select to enable the corresponding option for all the accounts in an organization. This includes new accounts that join the organization and those accounts that may have been suspended or removed from the organization. This also includes the delegated GuardDuty administrator account.

     Note

     It may take up to 24 hours to update the configuration for all member accounts.

   - **Auto-enable for new accounts (NEW)** – Select to enable GuardDuty or the optional protection plans for only new member accounts automatically when they join your organization.

   - **Do not enable (NONE)** – Select to prevent enabling the corresponding option for new accounts in your organization. In this case, the GuardDuty administrator account will manage each account individually.

When you update the auto-enable setting from `ALL` or `NEW` to `NONE`, this action doesn't disable the corresponding option for your existing accounts. This configuration will apply to the new accounts that join the organization. After you update the auto-enable settings, no new account will have the corresponding option as enabled.

> **Note**
>
> When a delegated GuardDuty administrator account opts out of an opt-in Region, even if your organization has the GuardDuty auto-enable configuration set to either new member accounts only (`NEW`) or all member accounts (`ALL`), GuardDuty cannot be enabled for any member account in the organization that currently has GuardDuty disabled. For information about the configuration of your member accounts, open **Accounts** in the GuardDuty console navigation pane or use the ListMembers API.

4. Choose **Save changes**.
5. (Optional) if you want to use the same preferences in each Region, update your preferences in each of the supported Regions separately.

   Some of the optional protection plans may not be available in all the AWS Regions where GuardDuty is available. For more information, see Regions and endpoints.

API/CLI

1. Run UpdateOrganizationConfiguration by using the credentials of the delegated GuardDuty administrator account, to automatically configure GuardDuty and optional protection plans in that Region for your organization. For information about the various auto-enable configurations, see autoEnableOrganizationMembers.

   To find the `detectorId` for your account and current Region, see the **Settings** page in the https://console.aws.amazon.com/guardduty/ console, or run the ListDetectors API.

   To set auto-enable preferences for any of the supported optional protection plans in your Region, follow the steps provided in the corresponding documentation sections of each protection plan.

2. You can validate the preferences for your organization in the current Region. Run describeOrganizationConfiguration. Make sure to specify the detector ID of the delegated GuardDuty administrator account.

   Note

   It may take up to 24 hours to update the configuration for all the member accounts.

3. Alternatively, run the following AWS CLI command to set the preferences to automatically enable or disable GuardDuty in that Region for new accounts (`NEW`) that join the organization, all the accounts (`ALL`), or none of the accounts (`NONE`) in the organization. For more information, see autoEnableOrganizationMembers. Based on your preference, you may need to replace `NEW` with `ALL` or `NONE`. If you configure the protection plan with `ALL`, the protection plan will also be enabled for the delegated GuardDuty administrator account. Make sure to specify the detector ID of the delegated GuardDuty administrator account that manages the organization configuration.

   To find the `detectorId` for your account and current Region, see the **Settings** page in the https://console.aws.amazon.com/guardduty/ console, or run the ListDetectors API.

```
aws guardduty update-organization-configuration --detector-
id 12abc34d567e8fa901bc2d34e56789f0 --auto-enable-organization-members=NEW
```

4. You can validate the preferences for your organization in the current Region. Run the following AWS CLI command by using the detector ID of the delegated GuardDuty administrator account.

```
aws guardduty describe-organization-configuration --detector-
id 12abc34d567e8fa901bc2d34e56789f0
```

(Recommended) repeat the previous steps in each Region by using the delegated GuardDuty administrator account detector ID.

Note

When a delegated GuardDuty administrator account opts out of an opt-in Region, even if your organization has the GuardDuty auto-enable configuration set to either new member accounts only (`NEW`) or all member accounts (`ALL`), GuardDuty cannot be enabled for any member account in the organization that currently has GuardDuty disabled. For information about the configuration of your member accounts, open **Accounts** in the GuardDuty console navigation pane or use the ListMembers API.

Adding members to the organization

As a delegated GuardDuty administrator account, you can add one or more AWS accounts to the GuardDuty organization. When you add an account as a GuardDuty member, it will automatically have GuardDuty enabled in that Region. There is an exception to the organization management account. Before the management account account gets added as a GuardDuty member, it must have GuardDuty enabled.

Choose a preferred method to add a member account to your GuardDuty organization.

Console

1. Open the GuardDuty console at https://console.aws.amazon.com/guardduty/.

   To sign in, use the delegated GuardDuty administrator account credentials.
2. In the navigation pane, choose **Accounts**.

   The accounts table displays all the member accounts that are active (not suspended AWS accounts) and may be associated with the delegated GuardDuty administrator account. If the member account is associated with the organization's administrator account, then the **Type** will be one of the following: **Via Organizations** or **By invitation**. If a member account is not associated with the organization's GuardDuty administrator account, the **Type** of this member account is **Not a member**.
3. Select one or more account IDs that you want to add as members. These account IDs must have the **Type** as **Via Organizations**.

   Accounts that are added through invitation are not a part of your organization. You can manage such accounts individually. For more information, see Managing accounts by invitation.
4. Choose the **Actions** dropdown, and then choose **Add member**. After you add this account as a member, the auto-enable GuardDuty configuration will apply. Based on the settings in Setting organization auto-enable preferences, the GuardDuty configuration of these accounts may change.

5. You can select the down arrow of the **Status** column to sort the accounts by the **Not a member** status and then choose each account that doesn't have GuardDuty enabled in the current Region.

   If none of the accounts listed in the accounts table have been added as a member yet, you can enable GuardDuty in the current Region for all organization accounts. Choose **Enable** in the banner at the top of the page. This action automatically turns on the **Auto-enable** GuardDuty configuration so that GuardDuty gets enabled for any new account that joins the organization.
6. Choose **Confirm** to add the accounts as members. This action also enables GuardDuty for all of the selected accounts. The **Status** for the accounts will change to **Enabled**.
7. (Recommended) Repeat these steps in each AWS Region. This ensures that the delegated GuardDuty administrator account can manage findings and other configurations for member accounts in all the Regions where you have GuardDuty enabled.

   The auto-enable feature enables GuardDuty for all future members of your organization. This allows your delegated GuardDuty administrator account to manage any new members that are created within or get added to the organization. When the number of member accounts reaches the limit of 50,000, the Auto-enable feature is automatically turned off. If you remove a member account and the total number of members decreases to fewer than 50,000, the Auto-enable feature turns back on.

API/CLI

- Run CreateMembers by using the credentials of the delegated GuardDuty administrator account.

  You must specify the regional detector ID of the delegated GuardDuty administrator account and the account details (AWS account IDs and corresponding email addresses) of the accounts that you want to add as GuardDuty members. You can create one or more members with this API operation.

  When you run CreateMembers in your organization, the auto-enable preferences for new members will apply as new member accounts join your organization. When you run CreateMembers with an existing member account, the organization configuration will also apply to the existing members. This might change the current configuration of the existing member accounts.

Run ListAccounts in the *AWS Organizations API Reference*, to view all the accounts in the AWS organization.

- Alternatively, you can use AWS Command Line Interface. Run the following AWS CLI command and make sure to use your own valid detector ID, AWS account ID, and the email address associated with the account ID.

  To find the `detectorId` for your account and current Region, see the **Settings** page in the https://console.aws.amazon.com/guardduty/ console, or run the ListDetectors API.

  ```
  aws guardduty create-members --detector-id 12abc34d567e8fa901bc2d34e56789f0
   --account-details AccountId=111122223333,Email=guardduty-member-
  name@amazon.com
  ```

  You can view a list of all organization members by running the following AWS CLI command:

  ```
  aws organizations list-accounts
  ```

After you add this account as a member, the auto-enable GuardDuty configuration will apply.

(Optional) Enable protection plans for existing member accounts

The following procedure includes steps to enable protection plans for existing member accounts by using the **Accounts** page. For steps to do this by using API or AWS CLI, see documents related to the specific protection plan.

You can enable protection plans for individual accounts through the **Accounts** page.

1. Open the GuardDuty console at https://console.aws.amazon.com/guardduty/.

   Use the delegated GuardDuty administrator account credentials.

2. In the navigation pane, choose **Accounts**.

3. Select one or more accounts for which you want to configure a protection plan. Repeat the following steps for each protection plan that you want to configure:

   a. Choose **Edit Protection Plans**.

b. From the list of protection plans, choose one protection plan that you want to configure.

c. Choose one of the actions that you want to perform for this protection plan, and then choose **Confirm**.

d. For the selected account, the column corresponding to the configured protection plan will show the updated configuration as **Enabled** or **Not enabled**.

Continually managing your member accounts within GuardDuty

As a delegated GuardDuty administrator account, you are responsible for maintaining the configuration of GuardDuty and its optional protection plans for all the accounts in your organization in each supported AWS Region. The following sections provide the options about maintaining the configuration status of GuardDuty or any of its optional protection plans:

To maintain the configuration status of your entire organization in each Region

- **Set auto-enable preferences for the entire organization by using GuardDuty console** – You can enable GuardDuty automatically for either all (`ALL`) the members in the organization or new (`NEW`) members joining the organization, or choose not to (`NONE`) auto-enable it any of the members in the organization.

  You can also configure the same or different settings for any of the protection plans within GuardDuty.

  It might take up to 24 hours to update the configuration for all member accounts in the organization.

- **Update auto-enable preferences by using API** – Run UpdateOrganizationConfiguration to automatically configure GuardDuty and its optional protection plans for the organization. When you run CreateMembers to add new member accounts in your organization, the configured settings will apply automatically. When you run CreateMembers with an existing member account, the organization configuration will also apply to the existing members. This might change the current configuration of the existing member accounts.

  To view all the accounts in your organization, run ListAccounts in the *AWS Organizations API Reference*.

To maintain the configuration status for member accounts individually in each Region

- To view all the accounts in your organization, run ListAccounts in the *AWS Organizations API Reference*.
- When you want selective member accounts to have a different configuration status, run UpdateMemberDetectors for each member account individually.

  You can use GuardDuty console to perform the same task by navigating to the **Accounts** page in the GuardDuty console.

  For information about enabling protection plans for individual accounts by using either console or API, see the configuring page for the corresponding protection plan.

Suspending GuardDuty for member account

As a delegated GuardDuty administrator account, you can suspend the GuardDuty service for a member account in your organization. If you do this, the member account stills stays in your GuardDuty organization. You can also re-enable GuardDuty for these member accounts at a later time. However, if you eventually want to disassociate (remove) this member account, then **after** following the steps in this section, you must follow the steps in Disassociating (removing) member account from administrator account.

When you suspend GuardDuty in a member account, you can expect the following changes:

- GuardDuty no longer monitors the security of the AWS environment, or generates new findings.
- The existing findings in the member account remain intact.
- A GuardDuty suspended member account does't incur any charges for GuardDuty.

  If the member account has enabled Malware Protection for S3 for one or more buckets in their account, then suspending GuardDuty doesn't impact the configuration of Malware Protection for S3. The member account will continue incurring the usage cost for Malware Protection for S3. For the member account to stop using Malware Protection for S3, they must disable this feature for the protected buckets. For more information, see Disabling Malware Protection for S3 for a protected bucket.

Choose a preferred method to suspend GuardDuty for a member account in your organization.

Console

1. Open the GuardDuty console at https://console.aws.amazon.com/guardduty/.

   To sign in, use the credentials of the delegated GuardDuty administrator account.
2. In the navigation pane, choose **Accounts**.
3. In the Accounts page, select one or more accounts for which you want to suspend GuardDuty.
4. Choose the **Actions** dropdown menu and then, choose **Suspend GuardDuty**.
5. Choose **Suspend GuardDuty** to confirm the selection.

   This will change the **Status** of the member account to **Disabled (suspended)**.

   Repeat the preceding steps in each additional Region where you want to disassociate or remove the member account.

API

1. To retrieve the member account account ID for which you want to suspend GuardDuty, use the ListMembers API. Include the `OnlyAssociated` parameter in your request. If you set this parameter's value to `true`, GuardDuty returns a `members` array that provides details about only those accounts that are currently GuardDuty members.

   Alternatively, you can use AWS Command Line Interface (AWS CLI) to run the following command:

   ```
   aws guardduty list-members --only-associated true --region us-east-1
   ```

   Replace *us-east-1* by the Region where you want to suspend GuardDuty for this account.
2. To suspend one or more GuardDuty member accounts, run StopMonitoringMembers to suspend GuardDuty for a member account.

   Alternatively, you can use AWS CLI to run the following command:

   ```
   aws guardduty stop-monitoring-members --detector-id
    12abc34d567e8fa901bc2d34EXAMPLE --account-ids 111122223333 --region us-east-1
   ```

Replace *us-east-1* by the Region where you want to suspend this account. If you have a list of account IDs that you want to remove, separate them by a space character.

If you further want to disassociate (remove) this member account, then follow the steps in Disassociating (removing) member account from administrator account.

Disassociating (removing) member account from administrator account

When you want to stop configuring the GuardDuty settings and accessing the data from a member account, remove that account as a GuardDuty member account. You can do it by disassociating (removing) that account from the GuardDuty administrator account.

When you disassociate a GuardDuty member account, GuardDuty remains enabled for the account in the current AWS Region. However, the account is disassociated from the delegated GuardDuty administrator account and the account becomes a standalone GuardDuty account. After you have disassociated the member account, it continues to show in the account inventory. GuardDuty doesn't notify the account's owner that you disassociated the account. You can add the account to your organization again at a later time.

Choose a preferred method to disassociate (remove) a member account from your organization.

Console

1. Open the GuardDuty console at https://console.aws.amazon.com/guardduty/.

   To sign in, use the credentials of the delegated GuardDuty administrator account.
2. In the navigation pane, choose **Accounts**.
3. In the **Accounts** table, you can remove an account that has **Type** as **Via Organizations** and **Status** as **Enabled**.

   Select one or more accounts with the same **Type** and **Status**.
4. From the **Actions** dropdown menu, choose **Disassociate account**.
5. Choose **Disassociate account** to confirm your selection.
6. The **Status** value for the selected accounts will change to **Not a member**. The **Via Organizations (Active/All)** count at the top right corner of the Accounts page will change to reflect the update.

Repeat the preceding steps in each additional Region where you want to disassociate the member account.

API

1. To retrieve the account ID for the member account that you want to remove, use the ListMembers API. Include the `OnlyAssociated` parameter in your request. If you set this parameter's value to `true`, GuardDuty returns a `members` array that provides details about only those accounts that are currently GuardDuty members.

   Alternatively, you can use AWS Command Line Interface (AWS CLI) to run the following command:

   ```
   aws guardduty list-members --only-associated true --region us-east-1
   ```

   Replace *us-east-1* by the Region where you want to remove this account.
2. To remove one or more GuardDuty member accounts, run DisassociateMembers to remove the member account that is associated with the administrator account.

   Alternatively, you can use AWS CLI to run the following command:

   ```
   aws guardduty disassociate-members --detector-id 12abc34d567e8fa901bc2d34EXAMPLE
    --account-ids 111122223333 --region us-east-1
   ```

   Replace *us-east-1* by the Region where you want to remove this account. If you have a list of account IDs that you want to remove, separate them by a space character.

Deleting member accounts from GuardDuty organization

As a delegated GuardDuty administrator account, after you have disassociated a member account and you no longer want to keep that member account in the GuardDuty organization, you can delete that member account from your GuardDuty organization. This member account will no longer appear in your account inventory. However, if GuardDuty was not suspended in this member account, the configuration of GuardDuty and dedicated protection plans remains the same. This account will now become a standalone account and can disable GuardDuty themselves.

This step will not delete the member account from your AWS organization.

Choose a preferred method to delete a member account from your GuardDuty organization.

Console

1. Open the GuardDuty console at https://console.aws.amazon.com/guardduty/.

   To sign in, use the credentials of the delegated GuardDuty administrator account.

2. In the navigation pane, choose **Accounts**.
3. In the **Accounts** table, you can remove an account that has **Type** as **Via Organizations** and **Status** as **Removed (disassociated)**.

   Select one or more accounts with the same **Type** and **Status**.

4. From the **Actions** dropdown menu, choose **Delete account**.
5. Choose **Delete accounts** to confirm your selection. The selected account member will no longer appear in your Accounts table.

   Repeat the preceding steps in each additional Region where you want to delete this member account.

API/CLI

1. To retrieve the account ID for the member account that you want to delete, use the ListMembers API. Include the `OnlyAssociated` parameter in your request. If you set this parameter's value to `false`, GuardDuty returns a `members` array that provides details about only those accounts that are currently disassociated GuardDuty members.

   Alternatively, you can use AWS Command Line Interface (AWS CLI) to run the following command:

   ```
   aws guardduty list-members --detector-id 12abc34d567e8fa901bc2d34EXAMPLE --only-associated="false" --region us-east-1
   ```

   Replace *12abc34d567e8fa901bc2d34EXAMPLE* with the delegated GuardDuty administrator account detector ID and *us-east-1* with the Region where you want to remove this account.

2. To delete one or more GuardDuty member accounts, run DeleteMembers to delete the member account from the GuardDuty organization.

Alternatively, you can use AWS CLI to run the following command:

```
aws guardduty delete-members --detector-id 12abc34d567e8fa901bc2d34EXAMPLE --
account-ids 111122223333 --region us-east-1
```

Replace *12abc34d567e8fa901bc2d34EXAMPLE* with the delegated GuardDuty administrator account detector ID and *us-east-1* by the Region where you want to remove this account. If you have a list of account IDs that you want to remove, separate them by a space character.

Changing the delegated GuardDuty administrator account

You can remove the delegated GuardDuty administrator account for your organization in each Region and then delegate a new administrator in each Region. To maintain the security posture for your organization's member accounts in a Region, you must have a delegated GuardDuty administrator account in that Region.

Note

Before you remove a delegated GuardDuty administrator account, you must disassociate all the member accounts associated with the delegated GuardDuty administrator account, and then delete them from the GuardDuty organization. For more information about these steps, see the following documents:

- Disassociating (removing) member account from administrator account
- Deleting member accounts from GuardDuty organization

Removing existing delegated GuardDuty administrator account

Step 1 - To remove existing delegated GuardDuty administrator account in each Region

1. As the existing delegated GuardDuty administrator account, list all the member accounts associated with your administrator account. Run ListMembers with `OnlyAssociated=false`.
2. If the auto-enable preference for GuardDuty or any of the optional protection plans is set to `ALL`, then run UpdateOrganizationConfiguration to update the organization configuration to

either `NEW` or `NONE`. This action will prevent an error when you disassociate all the member accounts in the next step.

3. Run [DisassociateMembers](#) to disassociate all the member accounts that are associated with the administrator account.
4. Run [DeleteMembers](#) to delete the associations between the administrator account and member accounts.
5. As the organization management account, run [DisableOrganizationAdminAccount](#) to remove the existing delegated GuardDuty administrator account.
6. Repeat these steps in each AWS Region where you have this delegated GuardDuty administrator account.

Step 2 - To de-register existing delegated GuardDuty administrator account in AWS Organizations (One-time global action)

- Run DeregisterDelegatedAdministrator in the *AWS Organizations API Reference*, to de-register the existing delegated GuardDuty administrator account in AWS Organizations.

  Alternatively, you can run the following AWS CLI command:

  ```
  aws organizations deregister-delegated-administrator --account-id 111122223333 --
  service-principal guardduty.amazonaws.com
  ```

  Make sure to replace *111122223333* with the existing delegated GuardDuty administrator account.

  After you de-register the old delegated GuardDuty administrator account, you can add it as a member account to the new delegated GuardDuty administrator account.

Designating a new delegated GuardDuty administrator account in each Region

1. Designate a new delegated GuardDuty administrator account in each Region by using your preferred access method - GuardDuty console, or API or AWS CLI. For more information, see Designating a delegated GuardDuty administrator account.
2. Run DescribeOrganizationConfiguration to view the current auto-enable configuration for your organization.

Important

Before you add any members to the new delegated GuardDuty administrator account, you must verify the auto-enable configuration for your organization. This configuration is specific to the new delegated GuardDuty administrator account and the selected Region, and doesn't relate to AWS Organizations. When you add (a new or an existing) organization member account under the new delegated GuardDuty administrator account, the auto-enable configuration of the new delegated GuardDuty administrator account will apply at the time of enabling GuardDuty or any of its optional protection plans.

Change the organization configuration for the new delegated GuardDuty administrator account by using your preferred access method - GuardDuty console, or API or AWS CLI. For more information, see Setting organization auto-enable preferences.

Managing GuardDuty accounts by invitation

To manage accounts outside of your organization, you can use the legacy invitation method. When you use this method, your account is designated as a administrator account when another account accepts your invitation to become a member account.

Note

GuardDuty recommends using AWS Organizations instead of GuardDuty invitations, to manage your member accounts. For more information, see Managing accounts with AWS Organizations.

If your account is not an administrator account, you can accept an invitation from another account. When you accept, your account becomes a member account. An AWS account cannot be a GuardDuty administrator account and member account at the same time.

When you accept an invitation from one account, you can't accept an invitation from another account. To accept an invitation from another account, you will first need to disassociate your account from the existing administrator account. Alternatively, the administrator account can also disassociate and remove your account from their organization.

Accounts associated by invitation have the same overall administrator account-to-member relationship as accounts associated by AWS Organizations, as described in Understanding the relationship between GuardDuty administrator account and member accounts. However, invitation administrator account users cannot enable GuardDuty on behalf of associated member accounts or view other non-member accounts within their AWS Organizations organization.

> **Important**
>
> Cross-regional data transfer may occur when GuardDuty creates member accounts using this method. In order to verify member accounts' email addresses, GuardDuty uses an email verification service that operates only in the US East (N. Virginia) Region.

Topics

- Adding accounts by invitation
- Consolidating GuardDuty administrator accounts under a single organization

Adding accounts by invitation

As an administrator account that already has GuardDuty enabled, you can add members to start using GuardDuty. After adding the members, you can invite them to join GuardDuty, and they can choose to respond to your invitation.

> **Note**
>
> GuardDuty recommends using AWS Organizations instead of GuardDuty invitations, to manage your member accounts. For more information, see Managing accounts with AWS Organizations.

Choose a preferred access method to add GuardDuty member accounts as a GuardDuty administrator account.

Console

Step 1 - Add an account

1. Open the GuardDuty console at https://console.aws.amazon.com/guardduty/.

2. In the navigation pane, choose **Accounts**.
3. Choose **Add accounts by invitation** in the top pane.
4. On the **Add member accounts** page, under **Enter account details**, enter the AWS account ID and email address associated with the account that you want to add.
5. To add another row to enter account details one at a time, choose **Add another account**. You can also choose **Upload .csv file with account details** to add accounts in bulk.

> **Important**
>
> The first line of your csv file must contain the header, as depicted in the following example – `Account ID,Email`. Each subsequent line must contain a single valid AWS account ID and its associated email address. The format of a row is valid if it contains only one AWS account ID and the associated email address separated by a comma.
>
> ```
> Account ID,Email
>                                  555555555555,user@example.com
> ```

6. After you have added all the accounts' details, choose **Next**. You can view the newly-added accounts in the Accounts table. The **Status** of these accounts will be **Invite not sent**. For information about sending an invite to one or more added accounts, see Step 2 - Invite an account.

Step 2 - Invite an account

1. Open the GuardDuty console at https://console.aws.amazon.com/guardduty/.
2. In the navigation pane, choose **Accounts**.
3. Select one or more accounts that you want to invite to Amazon GuardDuty.
4. Choose **Actions** dropdown menu and then choose **Invite**.
5. In the **Invitation to GuardDuty** dialog box, enter an (optional) invitation message.

   If the invited account does not have access to email, select the checkbox **Also send an email notification to the root user on the invitee's AWS account and generate an alert in the invitee's AWS Health Dashboard**.

6. Choose **Send invitation**. If the invitees have access to the specified email address they can view the invite by opening the GuardDuty console at https://console.aws.amazon.com/guardduty/.
7. When an invitee accepts the invite, the value in the **Status** column changes to **Invited**. For information about accepting an invite, see [Step 3 - Accept an invitation](#).

Step 3 - Accept an invitation

1. Open the GuardDuty console at https://console.aws.amazon.com/guardduty/.

   > **Important**
   >
   > You must enable GuardDuty before you can view or accept a membership invitation.

2. Do the following only if you haven't enabled GuardDuty yet; otherwise, you can skip this step and continue with the next step.

   If you haven't yet enabled GuardDuty, choose **Get Started** on the Amazon GuardDuty page.

   On the **Welcome to GuardDuty** page, choose **Enable GuardDuty**.
3. After you enable GuardDuty for your account, use the following steps to accept the membership invitation:
   a. In the navigation pane, choose **Settings**.
   b. Choose **Accounts**.
   c. On the **Accounts**, ensure to verify the owner of the account from which you accept the invitation. Turn on **Accept** to accept the membership invite.
4. After you accept the invite, your account becomes a GuardDuty member account. The account whose owner sent the invitation becomes the GuardDuty administrator account. The administrator account will know that you have accepted the invitation. The **Accounts** table in their GuardDuty account will get updated. The value in the **Status** column corresponding to your member account ID will change to **Enabled**. The administrator account owner can now view and manage GuardDuty and protection plan configurations on behalf of your account. The administrator account can also view and manage GuardDuty findings generated for your member account.

API/CLI

You can designate a GuardDuty administrator account, and create or add GuardDuty member accounts by invitation through the API operations. Run the following GuardDuty API operations in order to designate administrator account and member accounts in GuardDuty.

Complete the following procedure using the credentials of the AWS account that you want to designate as the GuardDuty administrator account.

Creating or adding member accounts

1. Run the CreateMembers API operation using the credentials of the AWS account that has GuardDuty enabled. This is the account that you want to be the administrator account GuardDuty account.

   You must specify the detector ID of the current AWS account and the account ID and email address of the accounts that you want to become GuardDuty members. You can create one or more members with this API operation.

   You can also use AWS Command Line Tools to designate a administrator account by running the following CLI command. Make sure to use your own valid detector ID, account ID, and email.

   To find the `detectorId` for your account and current Region, see the **Settings** page in the https://console.aws.amazon.com/guardduty/ console, or run the ListDetectors API.

   ```
   aws guardduty create-members --detector-id 12abc34d567e8fa901bc2d34e56789f0 --
   account-details AccountId=111122223333,Email=guardduty-member@organization.com
   ```

2. Run InviteMembers by using the credentials of the AWS account that has GuardDuty enabled. This is the account that you want to be the administrator account GuardDuty account.

   You must specify the detector ID of the current AWS account and the account IDs of the accounts that you want to become GuardDuty members. You can invite one or more members with this API operation.

> **Note**
>
> You can also specify an optional invitation message by using the `message` request parameter.

You can also use AWS Command Line Interface to designate member accounts by running the following command. Make sure to use your own valid detector ID and valid account IDs for the accounts you want to invite.

To find the `detectorId` for your account and current Region, see the **Settings** page in the https://console.aws.amazon.com/guardduty/ console, or run the ListDetectors API.

```
aws guardduty invite-members --detector-id 12abc34d567e8fa901bc2d34e56789f0 --
account-ids 111122223333
```

Accepting invitations

Complete the following procedure using the credentials of each AWS account that you want to designate as a GuardDuty member account.

1. Run the CreateDetector API operation for each AWS account that was invited to become a GuardDuty member account and that you want to accept an invitation.

   You must specify if the detector resource is to be enabled using the GuardDuty service. A detector must be created and enabled in order for GuardDuty to become operational. You must first enable GuardDuty before accepting an invitation.

   You can also do this by using AWS Command Line Tools using the following CLI command.

   ```
   aws guardduty create-detector --enable
   ```

2. Run the AcceptAdministratorInvitation API operation for each AWS account that you want to accept the membership invitation, using that account's credentials.

   You must specify the detector ID of this AWS account for the member account, the account ID of the administrator account that sent the invitation, and the invitation ID of the invitation that you are accepting. You can find the account ID of the administrator account in the invitation email or by using the ListInvitations operation of the API.

You can also accept an invitation using AWS Command Line Tools by running the following CLI command. Make sure to use a valid detector ID, administrator account ID, and an invitation ID.

To find the `detectorId` for your account and current Region, see the **Settings** page in the https://console.aws.amazon.com/guardduty/ console, or run the ListDetectors API.

```
aws guardduty accept-invitation --detector-id 12abc34d567e8fa901bc2d34e56789f0
 --administrator-id 444455556666 --invitation-
id 84b097800250d17d1872b34c4daadcf5
```

Consolidating GuardDuty administrator accounts under a single organization

GuardDuty recommends using association through AWS Organizations to manage member accounts under a delegated GuardDuty administrator account. You can use the example process outlined below to consolidate administrator account and member associated by invitation in an organization under a single GuardDuty delegated GuardDuty administrator account.

> **Note**
>
> GuardDuty recommends using AWS Organizations instead of GuardDuty invitations, to manage your member accounts. For more information, see Managing accounts with AWS Organizations.

Accounts that are already being managed by a delegated GuardDuty administrator account, or active member accounts that are associated with delegated GuardDuty administrator account can't be added to a different delegated GuardDuty administrator account. Each organization can have only one delegated GuardDuty administrator account per Region, and each member account can have only one delegated GuardDuty administrator account.

Choose a preferred access method to consolidate GuardDuty administrator accounts under a single delegated GuardDuty administrator account.

Console

1. Open the GuardDuty console at https://console.aws.amazon.com/guardduty/.

   To log in, use the credentials of the management account of the organization.
2. All the accounts for which you want to manage GuardDuty must be a part of your organization. For information about adding an account to your organization, see Inviting an AWS account to join your organization.
3. Make sure all the member accounts are associated with the account that you want to designate as the single delegated GuardDuty administrator account. Disassociate any member account that is still associated with the pre-existing administrator accounts.

   The following steps will help you disassociate member accounts from the pre-existing administrator account:

   a. Open the GuardDuty console at https://console.aws.amazon.com/guardduty/.
   b. To log in, use the credentials of the pre-existing administrator account.
   c. In the navigation pane, choose **Accounts**.
   d. On the **Accounts** page, select one or more accounts that you want to disassociate from the administrator account.
   e. Choose **Actions** and then choose **Disassociate account**.
   f. Choose **Confirm** to finalize the step.
4. Open the GuardDuty console at https://console.aws.amazon.com/guardduty/.

   To log in, use the management account credentials.
5. In the navigation pane, choose **Settings**. On the **Settings** page, designate the delegated GuardDuty administrator account for the organization.
6. Log in to the designated delegated GuardDuty administrator account.
7. Add members from the organization. For more information, see Managing GuardDuty accounts with AWS Organizations.

API/CLI

1. All the accounts for which you want to manage GuardDuty must be a part of your organization. For information about adding an account to your organization, see Inviting an AWS account to join your organization.

2. Make sure all the member accounts are associated with the account that you want to designate as the single delegated GuardDuty administrator account.

   a. Run DisassociateMembers to disassociate any member account that is still associated with the pre-existing administrator accounts.

   b. Alternatively, you can use AWS Command Line Interface to run the following command and replace *777777777777* with the detector ID of the pre-existing administrator account from which you want to disassociate the member account. Replace *666666666666* with the AWS account ID of the member account that you want to disassociate.

      ```
      aws guardduty disassociate-members --detector-id 777777777777 --account-
      ids 666666666666
      ```

3. Run EnableOrganizationAdminAccount to delegate an AWS account as the delegated GuardDuty administrator account.

   Alternatively, you can use AWS Command Line Interface to run the following command to delegate a delegated GuardDuty administrator account:

   ```
   aws guardduty enable-organization-admin-account --admin-account-id 777777777777
   ```

4. Add members from the organization. For more information, see Create or add member member accounts using API.

Important

To maximize the effectiveness of GuardDuty, a regional service, we recommend that you designate your delegated GuardDuty administrator account and add all your member accounts in every Region.

GuardDuty considerations for exporting member account details in CSV format

As a GuardDuty administrator account, you can export the member account details in a CSV format. These details include the member account ID, name, type (added by AWS Organizations or through invitation), and configuration status of GuardDuty and dedicated protection plans.

The **Export CSV** option is displayed on the GuardDuty **Accounts** page based on how you manage the multiple member accounts. By using the **Export CSV** option, you can identify which member accounts have a specific protection plan enabled.

The following list provides the criteria whether or not the **Export CSV** will be available on your GuardDuty **Accounts** page:

- You use only AWS Organizations to manage multiple member accounts and the total number of member accounts in your GuardDuty organization are up to 5,000.
- You use both AWS Organizations and invitations method, and the total number of member accounts in your GuardDuty organization are up to 5,000.

  In this scenario, the exported CSV will include whether a member account was added through AWS Organizations or by using invitation-based method.
- When you use only the invitation-based method to manage multiple member accounts, there is no **Export CSV** option.

GuardDuty finding types

A finding is a notification that GuardDuty generates when it detects an indication of a suspicious or malicious activity in your AWS account. GuardDuty generates a finding in an account that has enabled GuardDuty.

For information about important changes to the GuardDuty finding types, including newly added or retired finding types, see Document history for Amazon GuardDuty.

For information about finding types which are now retired, see Retired finding types.

GuardDuty EC2 finding types

The following findings are specific to Amazon EC2 resources and always have a Resource Type of `Instance`. The severity and details of the findings differ based on the Resource Role, which indicates whether the EC2 resource was the target of suspicious activity or the actor performing the activity.

The findings listed here include the data sources and models used to generate that finding type. For more information data sources and models see GuardDuty foundational data sources.

Notes

- EC2 finding instance details may be missing if the instance was already terminated, or if the underlying API call originated from an EC2 instance in a different Region.
- EC2 findings that use VPC flow logs as a data source do not support IPv6 traffic.

For all EC2 findings, it is recommended that you examine the resource in question to determine if it is behaving in an expected manner. If the activity is authorized, you can use Suppression Rules or Trusted IP lists to prevent false positive notifications for that resource. If the activity is unexpected, the security best practice is to assume the instance has been compromised and take the actions detailed in Remediating a potentially compromised Amazon EC2 instance.

Topics

- Backdoor:EC2/C&CActivity.B

- Backdoor:EC2/C&CActivity.B!DNS
- Backdoor:EC2/DenialOfService.Dns
- Backdoor:EC2/DenialOfService.Tcp
- Backdoor:EC2/DenialOfService.Udp
- Backdoor:EC2/DenialOfService.UdpOnTcpPorts
- Backdoor:EC2/DenialOfService.UnusualProtocol
- Backdoor:EC2/Spambot
- Behavior:EC2/NetworkPortUnusual
- Behavior:EC2/TrafficVolumeUnusual
- CryptoCurrency:EC2/BitcoinTool.B
- CryptoCurrency:EC2/BitcoinTool.B!DNS
- DefenseEvasion:EC2/UnusualDNSResolver
- DefenseEvasion:EC2/UnusualDoHActivity
- DefenseEvasion:EC2/UnusualDoTActivity
- Impact:EC2/AbusedDomainRequest.Reputation
- Impact:EC2/BitcoinDomainRequest.Reputation
- Impact:EC2/MaliciousDomainRequest.Reputation
- Impact:EC2/MaliciousDomainRequest.Custom
- Impact:EC2/PortSweep
- Impact:EC2/SuspiciousDomainRequest.Reputation
- Impact:EC2/WinRMBruteForce
- Recon:EC2/PortProbeEMRUnprotectedPort
- Recon:EC2/PortProbeUnprotectedPort
- Recon:EC2/Portscan
- Trojan:EC2/BlackholeTraffic
- Trojan:EC2/BlackholeTraffic!DNS
- Trojan:EC2/DGADomainRequest.B
- Trojan:EC2/DGADomainRequest.C!DNS
- Trojan:EC2/DNSDataExfiltration
- Trojan:EC2/DriveBySourceTraffic!DNS

- Trojan:EC2/DropPoint
- Trojan:EC2/DropPoint!DNS
- Trojan:EC2/PhishingDomainRequest!DNS
- UnauthorizedAccess:EC2/MaliciousIPCaller.Custom
- UnauthorizedAccess:EC2/MetadataDNSRebind
- UnauthorizedAccess:EC2/RDPBruteForce
- UnauthorizedAccess:EC2/SSHBruteForce
- UnauthorizedAccess:EC2/TorClient
- UnauthorizedAccess:EC2/TorRelay

Backdoor:EC2/C&CActivity.B

An EC2 instance is querying an IP that is associated with a known command and control server.

Default severity: High

- **Data source:** VPC flow logs

This finding informs you that the listed instance within your AWS environment is querying an IP associated with a known command and control (C&C) server. The listed instance might be compromised. Command and control servers are computers that issue commands to members of a botnet.

A botnet is a collection of internet-connected devices which might include PCs, servers, mobile devices, and Internet of Things devices, that are infected and controlled by a common type of malware. Botnets are often used to distribute malware and gather misappropriated information, such as credit card numbers. Depending on the purpose and structure of the botnet, the C&C server might also issue commands to begin a distributed denial of service (DDoS) attack.

> **Note**
>
> If the IP queried is log4j-related, then fields of the associated finding will include the following values:

- service.additionalInfo.threatListName = Amazon
- service.additionalInfo.threatName = Log4j Related

Remediation recommendations:

If this activity is unexpected, your instance may be compromised. For more information, see Remediating a potentially compromised Amazon EC2 instance.

Backdoor:EC2/C&CActivity.B!DNS

An EC2 instance is querying a domain name that is associated with a known command and control server.

Default severity: High

- **Data source:** DNS logs

This finding informs you that the listed instance within your AWS environment is querying a domain name associated with a known command and control (C&C) server. The listed instance might be compromised. Command and control servers are computers that issue commands to members of a botnet.

A botnet is a collection of internet-connected devices which might include PCs, servers, mobile devices, and Internet of Things devices, that are infected and controlled by a common type of malware. Botnets are often used to distribute malware and gather misappropriated information, such as credit card numbers. Depending on the purpose and structure of the botnet, the C&C server might also issue commands to begin a distributed denial of service (DDoS) attack.

Note

If the domain name queried is log4j-related, then the fields of the associated finding will include the following values:

- service.additionalInfo.threatListName = Amazon
- service.additionalInfo.threatName = Log4j Related

> **Note**
>
> To test how GuardDuty generates this finding type, you can make a DNS request from your instance (using `dig` for Linux or `nslookup` for Windows) against a test domain `guarddutyc2activityb.com`.

Remediation recommendations:

If this activity is unexpected, your instance may be compromised. For more information, see Remediating a potentially compromised Amazon EC2 instance.

Backdoor:EC2/DenialOfService.Dns

An EC2 instance is behaving in a manner that may indicate it is being used to perform a Denial of Service (DoS) attack using the DNS protocol.

Default severity: High

- **Data source:** VPC flow logs

This finding informs you that the listed EC2 instance within your AWS environment is generating a large volume of outbound DNS traffic. This may indicate that the listed instance is compromised and being used to perform denial-of-service (DoS) attacks using DNS protocol.

> **Note**
>
> This finding detects DoS attacks only against publicly routable IP addresses, which are primary targets of DoS attacks.

Remediation recommendations:

If this activity is unexpected, your instance may be compromised. For more information, see Remediating a potentially compromised Amazon EC2 instance.

Backdoor:EC2/DenialOfService.Tcp

An EC2 instance is behaving in a manner indicating it is being used to perform a Denial of Service (DoS) attack using the TCP protocol.

Default severity: High

- **Data source:** VPC flow logs

This finding informs you that the listed EC2 instance within your AWS environment is generating a large volume of outbound TCP traffic. This may indicate that the instance is compromised and being used to perform denial-of-service (DoS) attacks using TCP protocol.

> **Note**
>
> This finding detects DoS attacks only against publicly routable IP addresses, which are primary targets of DoS attacks.

Remediation recommendations:

If this activity is unexpected, your instance may be compromised. For more information, see Remediating a potentially compromised Amazon EC2 instance.

Backdoor:EC2/DenialOfService.Udp

An EC2 instance is behaving in a manner indicating it is being used to perform a Denial of Service (DoS) attack using the UDP protocol.

Default severity: High

- **Data source:** VPC flow logs

This finding informs you that the listed EC2 instance within your AWS environment is generating a large volume of outbound UDP traffic. This may indicate that the listed instance is compromised and being used to perform denial-of-service (DoS) attacks using UDP protocol.

> **Note**
>
> This finding detects DoS attacks only against publicly routable IP addresses, which are primary targets of DoS attacks.

Remediation recommendations:

If this activity is unexpected, your instance may be compromised. For more information, see Remediating a potentially compromised Amazon EC2 instance.

Backdoor:EC2/DenialOfService.UdpOnTcpPorts

An EC2 instance is behaving in a manner that may indicate it is being used to perform a Denial of Service (DoS) attack using the UDP protocol on a TCP port.

Default severity: High

- **Data source:** VPC flow logs

This finding informs you that the listed EC2 instance within your AWS environment is generating a large volume of outbound UDP traffic targeted to a port that is typically used for TCP communication. This may indicate that the listed instance is compromised and being used to perform a denial-of-service (DoS) attacks using UDP protocol on a TCP port.

> **Note**
>
> This finding detects DoS attacks only against publicly routable IP addresses, which are primary targets of DoS attacks.

Remediation recommendations:

If this activity is unexpected, your instance may be compromised. For more information, see Remediating a potentially compromised Amazon EC2 instance.

Backdoor:EC2/DenialOfService.UnusualProtocol

An EC2 instance is behaving in a manner that may indicate it is being used to perform a Denial of Service (DoS) attack using an unusual protocol.

Default severity: High

- **Data source:** VPC flow logs

This finding informs you that the listed EC2 instance in your AWS environment is generating a large volume of outbound traffic from an unusual protocol type that is not typically used by EC2 instances, such as Internet Group Management Protocol. This may indicate that the instance is compromised and is being used to perform denial-of-service (DoS) attacks using an unusual protocol. This finding detects DoS attacks only against publicly routable IP addresses, which are primary targets of DoS attacks.

Remediation recommendations:

If this activity is unexpected, your instance may be compromised. For more information, see Remediating a potentially compromised Amazon EC2 instance.

Backdoor:EC2/Spambot

An EC2 instance is exhibiting unusual behavior by communicating with a remote host on port 25.

Default severity: Medium

- **Data source:** VPC flow logs

This finding informs you that the listed EC2 instance in your AWS environment is communicating with a remote host on port 25. This behavior is unusual because this EC2 instance has no prior history of communications on port 25. Port 25 is traditionally used by mail servers for SMTP communications. This finding indicates your EC2 instance might be compromised for use in sending out spam.

Remediation recommendations:

If this activity is unexpected, your instance may be compromised. For more information, see Remediating a potentially compromised Amazon EC2 instance.

Behavior:EC2/NetworkPortUnusual

An EC2 instance is communicating with a remote host on an unusual server port.

Default severity: Medium

- **Data source:** VPC flow logs

This finding informs you that the listed EC2 instance in your AWS environment is behaving in a way that deviates from the established baseline. This EC2 instance has no prior history of communications on this remote port.

> **Note**
>
> If the EC2 instance communicated on port 389 or port 1389, then the associated finding severity will be modified to High, and the finding fields will include the following value:
>
> - service.additionalInfo.context = Possible log4j callback

Remediation recommendations:

If this activity is unexpected, your instance may be compromised. For more information, see Remediating a potentially compromised Amazon EC2 instance.

Behavior:EC2/TrafficVolumeUnusual

An EC2 instance is generating unusually large amounts of network traffic to a remote host.

Default severity: Medium

- **Data source:** VPC flow logs

This finding informs you that the listed EC2 instance in your AWS environment is behaving in a way that deviates from the established baseline. This EC2 instance has no prior history of sending this much traffic to this remote host.

Remediation recommendations:

If this activity is unexpected, your instance may be compromised. For more information, see [Remediating a potentially compromised Amazon EC2 instance](#).

CryptoCurrency:EC2/BitcoinTool.B

An EC2 instance is querying an IP address that is associated with cryptocurrency-related activity.

Default severity: High

- **Data source:** VPC flow logs

This finding informs you that the listed EC2 instance in your AWS environment is querying an IP Address that is associated with Bitcoin or other cryptocurrency-related activity. Bitcoin is a worldwide cryptocurrency and digital payment system that can be exchanged for other currencies, products, and services. Bitcoin is a reward for bitcoin-mining and is highly sought after by threat actors.

Remediation recommendations:

If you use this EC2 instance to mine or manage cryptocurrency, or this instance is otherwise involved in blockchain activity, this finding could be expected activity for your environment. If this is the case in your AWS environment, we recommend that you set up a suppression rule for this finding. The suppression rule should consist of two filter criteria. The first criteria should use the **Finding type** attribute with a value of `CryptoCurrency:EC2/BitcoinTool.B`. The second filter criteria should be the **Instance ID** of the instance involved in blockchain activity. To learn more about creating suppression rules see Suppression rules in GuardDuty.

If this activity is unexpected, your instance is likely compromised, see Remediating a potentially compromised Amazon EC2 instance.

CryptoCurrency:EC2/BitcoinTool.B!DNS

An EC2 instance is querying a domain name that is associated with cryptocurrency-related activity.

Default severity: High

- **Data source:** DNS logs

This finding informs you that the listed EC2 instance in your AWS environment is querying a domain name that is associated with Bitcoin or other cryptocurrency-related activity. Bitcoin is a worldwide cryptocurrency and digital payment system that can be exchanged for other currencies, products, and services. Bitcoin is a reward for bitcoin-mining and is highly sought after by threat actors.

Remediation recommendations:

If you use this EC2 instance to mine or manage cryptocurrency, or this instance is otherwise involved in blockchain activity, this finding could be expected activity for your environment. If this is the case in your AWS environment, we recommend that you set up a suppression rule for this finding. The suppression rule should consist of two filter criteria. The first criteria should use the **Finding type** attribute with a value of `CryptoCurrency:EC2/BitcoinTool.B!DNS`. The second filter criteria should be the **Instance ID** of the instance involved in blockchain activity. To learn more about creating suppression rules see Suppression rules in GuardDuty.

If this activity is unexpected, your instance is likely compromised, see Remediating a potentially compromised Amazon EC2 instance.

DefenseEvasion:EC2/UnusualDNSResolver

An Amazon EC2 instance is communicating with an unusual public DNS resolver.

Default severity: Medium

- **Data source:** VPC flow logs

This finding informs you that the listed Amazon EC2 instance in your AWS environment is behaving in a way that deviates from the baseline behavior. This EC2 instance has no recent history of communicating with this public DNS resolver. The **Unusual** field in the finding details panel in the GuardDuty console can provide information about the queried DNS resolver.

Remediation recommendations:

If this activity is unexpected, your instance may be compromised. For more information, see [Remediating a potentially compromised Amazon EC2 instance](#).

DefenseEvasion:EC2/UnusualDoHActivity

An Amazon EC2 instance is performing an unusual DNS over HTTPS (DoH) communication.

Default severity: Medium

- **Data source:** VPC flow logs

This finding informs you that the listed Amazon EC2 instance within your AWS environment is behaving in a way that deviates from the established baseline. This EC2 instance doesn't have any recent history of DNS over HTTPS (DoH) communications with this public DoH server. The **Unusual** field in the finding details can provide information about the queried DoH server.

Remediation recommendations:

If this activity is unexpected, your instance may be compromised. For more information, see [Remediating a potentially compromised Amazon EC2 instance](#).

DefenseEvasion:EC2/UnusualDoTActivity

An Amazon EC2 instance is performing an unusual DNS over TLS (DoT) communication.

Default severity: Medium

- **Data source:** VPC flow logs

This finding informs you that the listed EC2 instance in your AWS environment is behaving in a way that deviates from the established baseline. This EC2 instance doesn't have any recent history of

DNS over TLS (DoT) communications with this public DoT server. The **Unusual** field in the finding details panel can provide information about the queried DoT server.

Remediation recommendations:

If this activity is unexpected, your instance may be compromised. For more information, see Remediating a potentially compromised Amazon EC2 instance.

Impact:EC2/AbusedDomainRequest.Reputation

An EC2 instance is querying a low reputation domain name that is associated with known abused domains.

Default severity: Medium

- **Data source:** DNS logs

This finding informs you that the listed Amazon EC2 instance within your AWS environment is querying a low reputation domain name associated with known abused domains or IP addresses. Examples of abused domains are top level domain names (TLDs) and second-level domain names (2LDs) providing free subdomain registrations as well as dynamic DNS providers. Threat actors tend to use these services to register domains for free or at low costs. Low reputation domains in this category may also be expired domains resolving to a registrar's parking IP address and therefore may no longer be active. A parking IP is where a registrar directs traffic for domains that have not been linked to any service. The listed Amazon EC2 instance may be compromised as threat actors commonly use these registrar's or services for C&C and malware distribution.

Low reputation domains are based on a reputation score model. This model evaluates and ranks the characteristics of a domain to determine its likelihood of being malicious.

Remediation recommendations:

If this activity is unexpected, your instance may be compromised. For more information, see Remediating a potentially compromised Amazon EC2 instance.

Impact:EC2/BitcoinDomainRequest.Reputation

An EC2 instance is querying a low reputation domain name that is associated with cryptocurrency-related activity.

Default severity: High

- **Data source:** DNS logs

This finding informs you that the listed Amazon EC2 instance within your AWS environment is querying a low reputation domain name associated with Bitcoin or other cryptocurrency-related activity. Bitcoin is a worldwide cryptocurrency and digital payment system that can be exchanged for other currencies, products, and services. Bitcoin is a reward for bitcoin-mining and is highly sought after by threat actors.

Low reputation domains are based on a reputation score model. This model evaluates and ranks the characteristics of a domain to determine its likelihood of being malicious.

Remediation recommendations:

If you use this EC2 instance to mine or manage cryptocurrency, or this instance is otherwise involved in blockchain activity, this finding could represent expected activity for your environment. If this is the case in your AWS environment, we recommend that you set up a suppression rule for this finding. The suppression rule should consist of two filter criteria. The first criteria should use the **Finding type** attribute with a value of `Impact:EC2/BitcoinDomainRequest.Reputation`. The second filter criteria should be the **Instance ID** of the instance involved in blockchain activity. To learn more about creating suppression rules see Suppression rules in GuardDuty.

If this activity is unexpected, your instance is likely compromised, see Remediating a potentially compromised Amazon EC2 instance.

Impact:EC2/MaliciousDomainRequest.Reputation

An EC2 instance is querying a low reputation domain that is associated with known malicious domains.

Default severity: High

- **Data source:** DNS logs

This finding informs you that the listed Amazon EC2 instance within your AWS environment is querying a low reputation domain name associated with known malicious domains or IP addresses.

For example, domains may be associated with a known sinkhole IP address. Sinkholed domains are domains that were previously controlled by a threat actor, and requests made to them can indicate the instance is compromised. These domains may also be correlated with known malicious campaigns or domain generation algorithms.

Low reputation domains are based on a reputation score model. This model evaluates and ranks the characteristics of a domain to determine its likelihood of being malicious.

Remediation recommendations:

If this activity is unexpected, your instance may be compromised. For more information, see [Remediating a potentially compromised Amazon EC2 instance](#).

Impact:EC2/MaliciousDomainRequest.Custom

An EC2 instance is querying a domain on a custom threat entity list.

Default severity: Medium

- **Data source:** DNS logs

This finding informs you that the listed Amazon EC2 instance within your AWS environment is querying a domain name that is included in threat entity list that you uploaded and activated. In GuardDuty, a threat entity list consists of known malicious domain names and IP addresses. GuardDuty generates findings based on the activity associated with the uploaded threat entity list. You can view name of the threat entity list in the finding details.

Remediation recommendations:

If this activity is unexpected, your instance may be compromised. For more information, see [Remediating a potentially compromised Amazon EC2 instance](#).

Impact:EC2/PortSweep

An EC2 instance is probing a port on a large number of IP addresses.

Default severity: High

- **Data source:** VPC flow logs

This finding informs you the listed EC2 instance in your AWS environment is probing a port on a large number of publicly routable IP addresses. This type of activity is typically used to find vulnerable hosts to exploit. In the finding details panel in your GuardDuty console, only the most recent remote IP address gets displayed

Remediation recommendations:

If this activity is unexpected, your instance may be compromised. For more information, see Remediating a potentially compromised Amazon EC2 instance.

Impact:EC2/SuspiciousDomainRequest.Reputation

An EC2 instance is querying a low reputation domain name that is suspicious in nature due to its age, or low popularity.

Default severity: Low

- **Data source:** DNS logs

This finding informs you that the listed Amazon EC2 instance within your AWS environment is querying a low reputation domain name that is suspected of being malicious. noticed characteristics of this domain that were consistent with previously observed malicious domains, however, our reputation model was unable to definitively relate it to a known threat. These domains are typically newly observed or receive a low amount of traffic.

Low reputation domains are based on a reputation score model. This model evaluates and ranks the characteristics of a domain to determine its likelihood of being malicious.

Remediation recommendations:

If this activity is unexpected, your instance may be compromised. For more information, see Remediating a potentially compromised Amazon EC2 instance.

Impact:EC2/WinRMBruteForce

An EC2 instance is performing an outbound Windows Remote Management brute force attack.

Default severity: Low*

> **Note**
>
> This finding's severity is low if your EC2 instance was the target of a brute force attack. This finding's severity is high if your EC2 instance is the actor being used to perform the brute force attack.

- **Data source:** VPC flow logs

This finding informs you that the listed EC2 instance in your AWS environment is performing a Windows Remote Management (WinRM) brute force attack aimed at gaining access to the Windows Remote Management service on Windows-based systems.

Remediation recommendations:

If this activity is unexpected, your instance may be compromised. For more information, see [Remediating a potentially compromised Amazon EC2 instance](#).

Recon:EC2/PortProbeEMRUnprotectedPort

An EC2 instance has an unprotected EMR related port which is being probed by a known malicious host.

Default severity: High

- **Data source:** VPC flow logs

This finding informs you that an EMR related sensitive port on the listed EC2 instance that is part of a cluster in your AWS environment is not blocked by a security group, an access control list (ACL), or an on-host firewall such as Linux IPTables. This finding also informs that known scanners on the Internet are actively probing this port. Ports that can trigger this finding, such as port 8088 (YARN Web UI port), could potentially be used for remote code execution.

Remediation recommendations:

You should block open access to ports on clusters from the internet and restrict access only to specific IP addresses that require access to these ports. For more information see, Security Groups for EMR Clusters.

Recon:EC2/PortProbeUnprotectedPort

An EC2 instance has an unprotected port that is being probed by a known malicious host.

Default severity: Low*

> **Note**
>
> This finding's default severity is Low. However, if the port that is being probed, is used by Elasticsearch (9200 or 9300), the finding's severity is High.

- **Data source:** VPC flow logs

This finding informs you that a port on the listed EC2 instance in your AWS environment is not blocked by a security group, access control list (ACL), or an on-host firewall such as Linux IPTables, and that known scanners on the internet are actively probing it.

If the identified unprotected port is 22 or 3389 and you are using these ports to connect to your instance, you can still limit exposure by allowing access to these ports only to the IP addresses from your corporate network IP address space. To restrict access to port 22 on Linux, see Authorizing Inbound Traffic for Your Linux Instances. To restrict access to port 3389 on Windows, see Authorizing Inbound Traffic for Your Windows Instances.

GuardDuty doesn't generate this finding for ports 443 and 80.

Remediation recommendations:

There may be cases in which instances are intentionally exposed, for example if they are hosting web servers. If this is the case in your AWS environment, we recommend that you set up a suppression rule for this finding. The suppression rule should consist of two filter criteria. The first criteria should use the **Finding type** attribute with a value of `Recon:EC2/PortProbeUnprotectedPort`. The second filter criteria should match the instance or instances that serve as a bastion host. You can use either the **Instance image ID** attribute or the **Tag** value attribute, depending on which criteria is identifiable with the instances that host these tools. For more information about creating suppression rules see Suppression rules in GuardDuty.

If this activity is unexpected, your instance is likely compromised, see Remediating a potentially compromised Amazon EC2 instance.

Recon:EC2/Portscan

An EC2 instance is performing outbound port scans to a remote host.

Default severity: Medium

- **Data source:** VPC flow logs

This finding informs you that the listed EC2 instance in your AWS environment is engaged in a possible port scan attack because it is trying to connect to multiple ports over a short period of time. The purpose of a port scan attack is to locate open ports to discover which services the machine is running and to identify its operating system.

Remediation recommendations:

This finding can be a false positive when vulnerability assessment applications are deployed on EC2 instances in your environment because these applications conduct port scans to alert you about misconfigured open ports. If this is the case in your AWS environment, we recommend that you set up a suppression rule for this finding. The suppression rule should consist of two filter criteria. The first criteria should use the **Finding type** attribute with a value of `Recon:EC2/Portscan`. The second filter criteria should match the instance or instances that host these vulnerability assessment tools. You can use either the **Instance image ID** attribute or the **Tag** value attribute depending on which criteria are identifiable with the instances that host these tools. For more information about creating suppression rules see Suppression rules in GuardDuty.

If this activity is unexpected, your instance is likely compromised, see Remediating a potentially compromised Amazon EC2 instance.

Trojan:EC2/BlackholeTraffic

An EC2 instance is attempting to communicate with an IP address of a remote host that is a known black hole.

Default severity: Medium

- **Data source:** VPC flow logs

This finding informs you the listed EC2 instance in your AWS environment might be compromised because it is trying to communicate with an IP address of a black hole (or sink hole). Black holes are places in the network where incoming or outgoing traffic is silently discarded without informing the source that the data didn't reach its intended recipient. A black hole IP address specifies a host machine that is not running or an address to which no host has been assigned.

Remediation recommendations:

If this activity is unexpected, your instance may be compromised. For more information, see [Remediating a potentially compromised Amazon EC2 instance](#).

Trojan:EC2/BlackholeTraffic!DNS

An EC2 instance is querying a domain name that is being redirected to a black hole IP address.

Default severity: Medium

- **Data source:** DNS logs

This finding informs you the listed EC2 instance in your AWS environment might be compromised because it is querying a domain name that is being redirected to a black hole IP address. Black holes are places in the network where incoming or outgoing traffic is silently discarded without informing the source that the data didn't reach its intended recipient.

Remediation recommendations:

If this activity is unexpected, your instance may be compromised. For more information, see [Remediating a potentially compromised Amazon EC2 instance](#).

Trojan:EC2/DGADomainRequest.B

An EC2 instance is querying algorithmically generated domains. Such domains are commonly used by malware and could be an indication of a compromised EC2 instance.

Default severity: High

- **Data source:** DNS logs

This finding informs you that the listed EC2 instance in your AWS environment is trying to query domain generation algorithm (DGA) domains. Your EC2 instance might be compromised.

DGAs are used to periodically generate a large number of domain names that can be used as rendezvous points with their command and control (C&C) servers. Command and control servers are computers that issue commands to members of a botnet, which is a collection of internet-connected devices that are infected and controlled by a common type of malware. The large number of potential rendezvous points makes it difficult to effectively shut down botnets because infected computers attempt to contact some of these domain names every day to receive updates or commands.

> **Note**
>
> This finding is based on analysis of domain names using advanced heuristics and may identify new DGA domains that are not present in threat intelligence feeds.

Remediation recommendations:

If this activity is unexpected, your instance may be compromised. For more information, see Remediating a potentially compromised Amazon EC2 instance.

Trojan:EC2/DGADomainRequest.C!DNS

An EC2 instance is querying algorithmically generated domains. Such domains are commonly used by malware and could be an indication of a compromised EC2 instance.

Default severity: High

- **Data source:** DNS logs

This finding informs you that the listed EC2 instance in your AWS environment is trying to query domain generation algorithm (DGA) domains. Your EC2 instance might be compromised.

DGAs are used to periodically generate a large number of domain names that can be used as rendezvous points with their command and control (C&C) servers. Command and control servers are computers that issue commands to members of a botnet, which is a collection of internet-connected devices that are infected and controlled by a common type of malware. The large number of potential rendezvous points makes it difficult to effectively shut down botnets because infected computers attempt to contact some of these domain names every day to receive updates or commands.

> **Note**
>
> This finding is based on known DGA domains from GuardDuty's threat intelligence feeds.

Remediation recommendations:

If this activity is unexpected, your instance may be compromised. For more information, see [Remediating a potentially compromised Amazon EC2 instance](#).

Trojan:EC2/DNSDataExfiltration

An EC2 instance is exfiltrating data through DNS queries.

Default severity: High

- **Data source:** DNS logs

This finding informs you that the listed EC2 instance in your AWS environment is running malware that uses DNS queries for outbound data transfers. This type of data transfer is indicative of a compromised instance and could result in the exfiltration of data. DNS traffic is not typically blocked by firewalls. For example, malware in a compromised EC2 instance can encode data, (such as your credit card number), into a DNS query and send it to a remote DNS server that is controlled by an attacker.

Remediation recommendations:

If this activity is unexpected, your instance may be compromised. For more information, see Remediating a potentially compromised Amazon EC2 instance.

Trojan:EC2/DriveBySourceTraffic!DNS

An EC2 instance is querying a domain name of a remote host that is a known source of Drive-By download attacks.

Default severity: High

- **Data source:** DNS logs

This finding informs you that the listed EC2 instance in your AWS environment might be compromised because it is querying a domain name of a remote host that is a known source of drive-by download attacks. These are unintended downloads of computer software from the internet that can trigger an automatic installation of a virus, spyware, or malware.

Remediation recommendations:

If this activity is unexpected, your instance may be compromised. For more information, see Remediating a potentially compromised Amazon EC2 instance.

Trojan:EC2/DropPoint

An EC2 instance is attempting to communicate with an IP address of a remote host that is known to hold credentials and other stolen data captured by malware.

Default severity: Medium

- **Data source:** VPC flow logs

This finding informs you that an EC2 instance in your AWS environment is trying to communicate with an IP address of a remote host that is known to hold credentials and other stolen data captured by malware.

Remediation recommendations:

If this activity is unexpected, your instance may be compromised. For more information, see [Remediating a potentially compromised Amazon EC2 instance](#).

Trojan:EC2/DropPoint!DNS

An EC2 instance is querying a domain name of a remote host that is known to hold credentials and other stolen data captured by malware.

Default severity: Medium

- **Data source:** DNS logs

This finding informs you that an EC2 instance in your AWS environment is querying a domain name of a remote host that is known to hold credentials and other stolen data captured by malware.

Remediation recommendations:

If this activity is unexpected, your instance may be compromised. For more information, see [Remediating a potentially compromised Amazon EC2 instance](#).

Trojan:EC2/PhishingDomainRequest!DNS

An EC2 instance is querying domains involved in phishing attacks. Your EC2 instance might be compromised.

Default severity: High

- **Data source:** DNS logs

This finding informs you that there is an EC2 instance in your AWS environment that is trying to query a domain involved in phishing attacks. Phishing domains are set up by someone posing as a legitimate institution in order to induce individuals to provide sensitive data, such as personally identifiable information, banking and credit card details, and passwords. Your EC2 instance may be trying to retrieve sensitive data stored on a phishing website, or it may be attempting to set up a phishing website. Your EC2 instance might be compromised.

Remediation recommendations:

If this activity is unexpected, your instance may be compromised. For more information, see Remediating a potentially compromised Amazon EC2 instance.

UnauthorizedAccess:EC2/MaliciousIPCaller.Custom

An EC2 instance is making connections to an IP address on a custom threat list.

Default severity: Medium

- **Data source:** VPC flow logs

This finding informs you that an EC2 instance in your AWS environment is communicating with an IP address included on a threat list that you uploaded. In GuardDuty, a threat list consists of known malicious IP addresses. GuardDuty generates findings based on uploaded threat lists. The threat list used to generate this finding will be listed in the finding's details.

Remediation recommendations:

If this activity is unexpected, your instance may be compromised. For more information, see Remediating a potentially compromised Amazon EC2 instance.

UnauthorizedAccess:EC2/MetadataDNSRebind

An EC2 instance is performing DNS lookups that resolve to the instance metadata service.

Default severity: High

- **Data source:** DNS logs

This finding informs you that an EC2 instance in your AWS environment is querying a domain that resolves to the EC2 metadata IP address (169.254.169.254). A DNS query of this kind may indicate that the instance is a target of a DNS rebinding technique. This technique can be used to obtain metadata from an EC2 instance, including the IAM credentials associated with the instance.

DNS rebinding involves tricking an application running on the EC2 instance to load return data from a URL, where the domain name in the URL resolves to the EC2 metadata IP address (169.254.169.254). This causes the application to access EC2 metadata and possibly make it available to the attacker.

It is possible to access EC2 metadata using DNS rebinding only if the EC2 instance is running a vulnerable application that allows injection of URLs, or if someone accesses the URL in a web browser running on the EC2 instance.

Remediation recommendations:

In response to this finding, you should evaluate if there is a vulnerable application running on the EC2 instance, or if someone used a browser to access the domain identified in the finding. If the root cause is a vulnerable application, you should fix the vulnerability. If someone browsed the identified domain, you should block the domain or prevent users from accessing it. If you determine this finding was related to either case above, revoke the session associated with the EC2 instance.

Some AWS customers intentionally map the metadata IP address to a domain name on their authoritative DNS servers. If this is the case in your environment, we recommend that you set up a suppression rule for this finding. The suppression rule should consist of two filter criteria. The first criteria should use the **Finding type** attribute with a value of `UnauthorizedAccess:EC2/MetaDataDNSRebind`. The second filter criteria should be **DNS request domain** and the value should match the domain you have mapped to the metadata IP address (169.254.169.254). For more information on creating suppression rules see Suppression rules in GuardDuty.

UnauthorizedAccess:EC2/RDPBruteForce

An EC2 instance has been involved in RDP brute force attacks.

Default severity: Low*

> **Note**
>
> This finding's severity is low if your EC2 instance was the target of a brute force attack. This finding's severity is high if your EC2 instance is the actor being used to perform the brute force attack.

- **Data source:** VPC flow logs

This finding informs you that an EC2 instance in your AWS environment was involved in a brute force attack aimed at obtaining passwords to RDP services on Windows-based systems. This can indicate unauthorized access to your AWS resources.

Remediation recommendations:

If your instance's **Resource Role** is `ACTOR`, this indicates your instance has been used to perform RDP brute force attacks. Unless this instance has a legitimate reason to be contacting the IP address listed as the `Target`, it is recommended that you assume your instance has been compromised and take the actions listed in Remediating a potentially compromised Amazon EC2 instance.

If your instance's **Resource Role** is `TARGET`, this finding can be remediated by securing your RDP port to only trusted IPs through Security Groups, ACLs, or firewalls. For more information see Tips for securing your EC2 instances (Linux).

UnauthorizedAccess:EC2/SSHBruteForce

An EC2 instance has been involved in SSH brute force attacks.

Default severity: Low*

> **Note**
>
> This finding's severity is low if a brute force attack is aimed at one of your EC2 instances. This finding's severity is high if your EC2 instance is being used to perform the brute force attack.

- **Data source:** VPC flow logs

This finding informs you that an EC2 instance in your AWS environment was involved in a brute force attack aimed at obtaining passwords to SSH services on Linux-based systems. This can indicate unauthorized access to your AWS resources.

Note

This finding is generated only through monitoring traffic on port 22. If your SSH services are configured to use other ports, this finding is not generated.

Remediation recommendations:

If the target of the brute force attempt is a bastion host, this may represent expected behavior for your AWS environment. If this is the case, we recommend that you set up a suppression rule for this finding. The suppression rule should consist of two filter criteria. The first criteria should use the **Finding type** attribute with a value of `UnauthorizedAccess:EC2/SSHBruteForce`. The second filter criteria should match the instance or instances that serve as a bastion host. You can use either the **Instance image ID** attribute or the **Tag** value attribute depending on which criteria is identifiable with the instances that host these tools. For more information about creating suppression rules see Suppression rules in GuardDuty.

If this activity is not expected for your environment and your instance's **Resource Role** is `TARGET`, this finding can be remediated by securing your SSH port to only trusted IPs through Security Groups, ACLs, or firewalls. For more information, see Tips for securing your EC2 instances (Linux).

If your instance's **Resource Role** is `ACTOR`, this indicates the instance has been used to perform SSH brute force attacks. Unless this instance has a legitimate reason to be contacting the IP address listed as the `Target`, it is recommended that you assume your instance has been compromised and take the actions listed in Remediating a potentially compromised Amazon EC2 instance.

UnauthorizedAccess:EC2/TorClient

Your EC2 instance is making connections to a Tor Guard or an Authority node.

Default severity: High

- **Data source:** VPC flow logs

This finding informs you that an EC2 instance in your AWS environment is making connections to a Tor Guard or an Authority node. Tor is software for enabling anonymous communication. Tor Guards and Authority nodes act as initial gateways into a Tor network. This traffic can indicate that this EC2 instance has been compromised and is acting as a client on a Tor network. This finding may indicate unauthorized access to your AWS resources with the intent of hiding the attacker's true identity.

Remediation recommendations:

If this activity is unexpected, your instance may be compromised. For more information, see Remediating a potentially compromised Amazon EC2 instance.

UnauthorizedAccess:EC2/TorRelay

Your EC2 instance is making connections to a Tor network as a Tor relay.

Default severity: High

- **Data source:** VPC flow logs

This finding informs you that an EC2 instance in your AWS environment is making connections to a Tor network in a manner that suggests that it's acting as a Tor relay. Tor is software for enabling anonymous communication. Tor increases anonymity of communication by forwarding the client's possibly illicit traffic from one Tor relay to another.

Remediation recommendations:

If this activity is unexpected, your instance may be compromised. For more information, see Remediating a potentially compromised Amazon EC2 instance.

GuardDuty IAM finding types

The following findings are specific to IAM entities and access keys and always have a **Resource Type** of `AccessKey`. The severity and details of the findings differ based on the finding type.

The findings listed here include the data sources and models used to generate that finding type. For more information, see GuardDuty foundational data sources.

For all IAM-related findings, we recommend that you examine the entity in question and ensure that their permissions follow the best practice of least privilege. If the activity is unexpected, the credentials may be compromised. For information about remediating findings, see Remediating potentially compromised AWS credentials.

Topics

- CredentialAccess:IAMUser/AnomalousBehavior
- DefenseEvasion:IAMUser/AnomalousBehavior
- Discovery:IAMUser/AnomalousBehavior
- Exfiltration:IAMUser/AnomalousBehavior
- Impact:IAMUser/AnomalousBehavior
- InitialAccess:IAMUser/AnomalousBehavior
- PenTest:IAMUser/KaliLinux
- PenTest:IAMUser/ParrotLinux
- PenTest:IAMUser/PentooLinux
- Persistence:IAMUser/AnomalousBehavior
- Policy:IAMUser/RootCredentialUsage
- Policy:IAMUser/ShortTermRootCredentialUsage
- PrivilegeEscalation:IAMUser/AnomalousBehavior
- Recon:IAMUser/MaliciousIPCaller
- Recon:IAMUser/MaliciousIPCaller.Custom
- Recon:IAMUser/TorIPCaller
- Stealth:IAMUser/CloudTrailLoggingDisabled
- Stealth:IAMUser/PasswordPolicyChange
- UnauthorizedAccess:IAMUser/ConsoleLoginSuccess.B
- UnauthorizedAccess:IAMUser/InstanceCredentialExfiltration.InsideAWS
- UnauthorizedAccess:IAMUser/InstanceCredentialExfiltration.OutsideAWS
- UnauthorizedAccess:IAMUser/MaliciousIPCaller
- UnauthorizedAccess:IAMUser/MaliciousIPCaller.Custom
- UnauthorizedAccess:IAMUser/TorIPCaller

CredentialAccess:IAMUser/AnomalousBehavior

An API used to gain access to an AWS environment was invoked in an anomalous way.

Default severity: Medium

- **Data source:** CloudTrail management event

This finding informs you that an anomalous API request was observed in your account. This finding may include a single API or a series of related API requests made in proximity by a single user identity. The API observed is commonly associated with the credential access stage of an attack when an adversary is attempting to collect passwords, usernames, and access keys for your environment. The APIs in this category are `GetPasswordData`, `GetSecretValue`, `BatchGetSecretValue`, and `GenerateDbAuthToken`.

This API request was identified as anomalous by GuardDuty's anomaly detection machine learning (ML) model. The ML model evaluates all API requests in your account and identifies anomalous events that are associated with techniques used by adversaries. The ML model tracks various factors of the API request, such as, the user that made the request, the location the request was made from, and the specific API that was requested. Details on which factors of the API request are unusual for the user identity that invoked the request can be found in the finding details.

Remediation recommendations:

If this activity is unexpected, your credentials may be compromised. For more information, see Remediating potentially compromised AWS credentials.

DefenseEvasion:IAMUser/AnomalousBehavior

An API used to evade defensive measures was invoked in an anomalous way.

Default severity: Medium

- **Data source:** CloudTrail management event

This finding informs you that an anomalous API request was observed in your account. This finding may include a single API or a series of related API requests made in proximity by a single user identity. The API observed is commonly associated with defense evasion tactics where an adversary is trying to cover their tracks and avoid detection. APIs in this category are typically delete, disable, or stop operations, such as, `DeleteFlowLogs`, `DisableAlarmActions`, or `StopLogging`.

This API request was identified as anomalous by GuardDuty's anomaly detection machine learning (ML) model. The ML model evaluates all API requests in your account and identifies anomalous events that are associated with techniques used by adversaries. The ML model tracks various factors of the API request, such as, the user that made the request, the location the request was made from, and the specific API that was requested. Details on which factors of the API request are unusual for the user identity that invoked the request can be found in the finding details.

Remediation recommendations:

If this activity is unexpected, your credentials may be compromised. For more information, see Remediating potentially compromised AWS credentials.

Discovery:IAMUser/AnomalousBehavior

An API commonly used to discover resources was invoked in an anomalous way.

Default severity: Low

- **Data source:** CloudTrail management event

This finding informs you that an anomalous API request was observed in your account. This finding may include a single API or a series of related API requests made in proximity by a single user identity. The API observed is commonly associated with the discovery stage of an attack when an adversary is gathering information to determine if your AWS environment is susceptible to a broader attack. APIs in this category are typically get, describe, or list operations, such as, `DescribeInstances`, `GetRolePolicy`, or `ListAccessKeys`.

This API request was identified as anomalous by GuardDuty's anomaly detection machine learning (ML) model. The ML model evaluates all API requests in your account and identifies anomalous events that are associated with techniques used by adversaries. The ML model tracks various factors of the API request, such as, the user that made the request, the location the request was made from, and the specific API that was requested. Details on which factors of the API request are unusual for the user identity that invoked the request can be found in the finding details.

Remediation recommendations:

If this activity is unexpected, your credentials may be compromised. For more information, see Remediating potentially compromised AWS credentials.

Exfiltration:IAMUser/AnomalousBehavior

An API commonly used to collect data from an AWS environment was invoked in an anomalous way.

Default severity: High

- **Data source:** CloudTrail management event

This finding informs you that an anomalous API request was observed in your account. This finding may include a single API or a series of related API requests made in proximity by a single user identity. The API observed is commonly associated with exfiltration tactics where an adversary is trying to collect data from your network using packaging and encryption to avoid detection. APIs for this finding type are management (control-plane) operations only and are typically related to S3, snapshots, and databases, such as, `PutBucketReplication`, `CreateSnapshot`, or `RestoreDBInstanceFromDBSnapshot`.

This API request was identified as anomalous by GuardDuty's anomaly detection machine learning (ML) model. The ML model evaluates all API requests in your account and identifies anomalous events that are associated with techniques used by adversaries. The ML model tracks various factors of the API request, such as, the user that made the request, the location the request was made from, and the specific API that was requested. Details on which factors of the API request are unusual for the user identity that invoked the request can be found in the finding details.

Remediation recommendations:

If this activity is unexpected, your credentials may be compromised. For more information, see Remediating potentially compromised AWS credentials.

Impact:IAMUser/AnomalousBehavior

An API commonly used to tamper with data or processes in an AWS environment was invoked in an anomalous way.

Default severity: High

- **Data source:** CloudTrail management event

This finding informs you that an anomalous API request was observed in your account. This finding may include a single API or a series of related API requests made in proximity by a single user identity. The API observed is commonly associated with impact tactics where an adversary is trying to disrupt operations and manipulate, interrupt, or destroy data in your account. APIs for this finding type are typically delete, update, or put operations, such as, `DeleteSecurityGroup`, `UpdateUser`, or `PutBucketPolicy`.

This API request was identified as anomalous by GuardDuty's anomaly detection machine learning (ML) model. The ML model evaluates all API requests in your account and identifies anomalous events that are associated with techniques used by adversaries. The ML model tracks various factors of the API request, such as, the user that made the request, the location the request was made from, and the specific API that was requested. Details on which factors of the API request are unusual for the user identity that invoked the request can be found in the finding details.

Remediation recommendations:

If this activity is unexpected, your credentials may be compromised. For more information, see Remediating potentially compromised AWS credentials.

InitialAccess:IAMUser/AnomalousBehavior

An API commonly used to gain unauthorized access to an AWS environment was invoked in an anomalous way.

Default severity: Medium

- **Data source:** CloudTrail management event

This finding informs you that an anomalous API request was observed in your account. This finding may include a single API or a series of related API requests made in proximity by a single user identity. The API observed is commonly associated with the initial access stage of an attack when an adversary is attempting to establish access to your environment. APIs in this category are typically get token, or session operations, such as, `StartSession`, or `GetAuthorizationToken`.

This API request was identified as anomalous by GuardDuty's anomaly detection machine learning (ML) model. The ML model evaluates all API requests in your account and identifies anomalous events that are associated with techniques used by adversaries. The ML model tracks various factors of the API request, such as, the user that made the request, the location the request was made from, and the specific API that was requested. Details on which factors of the API request are unusual for the user identity that invoked the request can be found in the finding details.

Remediation recommendations:

If this activity is unexpected, your credentials may be compromised. For more information, see Remediating potentially compromised AWS credentials.

PenTest:IAMUser/KaliLinux

An API was invoked from a Kali Linux machine.

Default severity: Medium

- **Data source:** CloudTrail management event

This finding informs you that a machine running Kali Linux is making API calls using credentials that belong to the listed AWS account in your environment. Kali Linux is a popular penetration testing tool that security professionals use to identify weaknesses in EC2 instances that require patching. Attackers also use this tool to find EC2 configuration weaknesses and gain unauthorized access to your AWS environment.

Remediation recommendations:

If this activity is unexpected, your credentials may be compromised. For more information, see Remediating potentially compromised AWS credentials.

PenTest:IAMUser/ParrotLinux

An API was invoked from a Parrot Security Linux machine.

Default severity: Medium

- **Data source:** CloudTrail management event

This finding informs you that a machine running Parrot Security Linux is making API calls using credentials that belong to the listed AWS account in your environment. Parrot Security Linux is a popular penetration testing tool that security professionals use to identify weaknesses in EC2 instances that require patching. Attackers also use this tool to find EC2 configuration weaknesses and gain unauthorized access to your AWS environment.

Remediation recommendations:

If this activity is unexpected, your credentials may be compromised. For more information, see Remediating potentially compromised AWS credentials.

PenTest:IAMUser/PentooLinux

An API was invoked from a Pentoo Linux machine.

Default severity: Medium

- **Data source:** CloudTrail management event

This finding informs you that a machine running Pentoo Linux is making API calls using credentials that belong to the listed AWS account in your environment. Pentoo Linux is a popular penetration testing tool that security professionals use to identify weaknesses in EC2 instances that require patching. Attackers also use this tool to find EC2 configuration weaknesses and gain unauthorized access to your AWS environment.

Remediation recommendations:

If this activity is unexpected, your credentials may be compromised. For more information, see Remediating potentially compromised AWS credentials.

Persistence:IAMUser/AnomalousBehavior

An API commonly used to maintain unauthorized access to an AWS environment was invoked in an anomalous way.

Default severity: Medium

- **Data source:** CloudTrail management event

This finding informs you that an anomalous API request was observed in your account. This finding may include a single API or a series of related API requests made in proximity by a single user identity. The API observed is commonly associated with persistence tactics where an adversary has gained access to your environment and is attempting to maintain that access. APIs in this category are typically create, import, or modify operations, such as, `CreateAccessKey`, `ImportKeyPair`, or `ModifyInstanceAttribute`.

This API request was identified as anomalous by GuardDuty's anomaly detection machine learning (ML) model. The ML model evaluates all API requests in your account and identifies anomalous events that are associated with techniques used by adversaries. The ML model tracks various factors of the API request, such as, the user that made the request, the location the request was made from, and the specific API that was requested. Details on which factors of the API request are unusual for the user identity that invoked the request can be found in the finding details.

Remediation recommendations:

If this activity is unexpected, your credentials may be compromised. For more information, see Remediating potentially compromised AWS credentials.

Policy:IAMUser/RootCredentialUsage

An API was invoked using root user sign-in credentials.

Default severity: Low

- **Data source:** CloudTrail management events or CloudTrail data events for S3

This finding informs you that the root user sign-in credentials of the listed AWS account in your environment are being used to make requests to AWS services. It is recommended that users never use root user sign-in credentials to access AWS services. Instead, AWS services should be accessed using least privilege temporary credentials from AWS Security Token Service (STS). For situations where AWS STS is not supported, IAM user credentials are recommended. For more information, see IAM Best Practices.

> **Note**
>
> If S3 Protection is enabled for the account, then this finding may be generated in response to the attempts to run S3 data plane operations on Amazon S3 resources by using the root user sign-in credentials of the AWS account. The API call used will be listed in the finding details. If S3 Protection is not enabled, then this finding can only be triggered by Event log APIs. For more information about S3 Protection, see S3 Protection.

Remediation recommendations:

If this activity is unexpected, your credentials may be compromised. For more information, see Remediating potentially compromised AWS credentials.

Policy:IAMUser/ShortTermRootCredentialUsage

An API was invoked by using restricted root user credentials.

Default severity: Low

- **Data source:** AWS CloudTrail management events or AWS CloudTrail data events for S3

This finding informs you that restricted user credentials created for the listed AWS account in your environment, are being used to make requests to AWS services. It is recommended to use root user credentials only for those tasks that require root user credentials.

When possible, access the AWS services by using least privilege IAM roles with temporary credentials from AWS Security Token Service (AWS STS). For scenarios where AWS STS is not supported, the best practice is to use IAM user credentials. For more information, see Security best practices in IAM and Root user best practices for your AWS account in the *IAM User Guide*.

Remediation recommendations:

If this activity is unexpected, your credentials may be compromised. For more information, see Remediating potentially compromised AWS credentials.

PrivilegeEscalation:IAMUser/AnomalousBehavior

An API commonly used to obtain high-level permissions to an AWS environment was invoked in an anomalous way.

Default severity: Medium

- **Data source:** CloudTrail management events

This finding informs you that an anomalous API request was observed in your account. This finding may include a single API or a series of related API requests made in proximity by a single user identity. The API observed is commonly associated with privilege escalation tactics where an adversary is attempting to gain higher-level permissions to an environment. APIs in this category typically involve operations that change IAM policies, roles, and users, such as, `AssociateIamInstanceProfile`, `AddUserToGroup`, or `PutUserPolicy`.

This API request was identified as anomalous by GuardDuty's anomaly detection machine learning (ML) model. The ML model evaluates all API requests in your account and identifies anomalous events that are associated with techniques used by adversaries. The ML model tracks various factors of the API request, such as, the user that made the request, the location the request was made from, and the specific API that was requested. Details on which factors of the API request are unusual for the user identity that invoked the request can be found in the finding details.

Remediation recommendations:

If this activity is unexpected, your credentials may be compromised. For more information, see Remediating potentially compromised AWS credentials.

Recon:IAMUser/MaliciousIPCaller

An API was invoked from a known malicious IP address.

Default severity: Medium

- **Data source:** CloudTrail management events

This finding informs you that an API operation that can list or describe AWS resources in an account within your environment was invoked from an IP address that is included on a threat list. An attacker may use stolen credentials to perform this type of reconnaissance of your AWS resources in order to find more valuable credentials or determine the capabilities of the credentials they already have.

Remediation recommendations:

If this activity is unexpected, your credentials may be compromised. For more information, see Remediating potentially compromised AWS credentials.

Recon:IAMUser/MaliciousIPCaller.Custom

An API was invoked from a known malicious IP address.

Default severity: Medium

- **Data source:** CloudTrail management events

This finding informs you that an API operation that can list or describe AWS resources in an account within your environment was invoked from an IP address that is included on a custom threat list. The threat list used will be listed in the finding's details. An attacker might use stolen credentials to perform this type of reconnaissance of your AWS resources in order to find more valuable credentials or determine the capabilities of the credentials they already have.

Remediation recommendations:

If this activity is unexpected, your credentials may be compromised. For more information, see Remediating potentially compromised AWS credentials.

Recon:IAMUser/TorIPCaller

An API was invoked from a Tor exit node IP address.

Default severity: Medium

- **Data source:** CloudTrail management events

This finding informs you that an API operation that can list or describe AWS resources in an account within your environment was invoked from a Tor exit node IP address. Tor is software for enabling anonymous communication. It encrypts and randomly bounces communications through relays between a series of network nodes. The last Tor node is called the exit node. An attacker would use Tor to mask their true identity.

Remediation recommendations:

If this activity is unexpected, your credentials may be compromised. For more information, see Remediating potentially compromised AWS credentials.

Stealth:IAMUser/CloudTrailLoggingDisabled

AWS CloudTrail logging was disabled.

Default severity: Low

- **Data source:** CloudTrail management events

This finding informs you that a CloudTrail trail within your AWS environment was disabled. This can be an attacker's attempt to disable logging to cover their tracks by eliminating any trace of their activity while gaining access to your AWS resources for malicious purposes. This finding can be triggered by a successful deletion or update of a trail. This finding can also be triggered by a successful deletion of an S3 bucket that stores the logs from a trail that is associated with GuardDuty.

Remediation recommendations:

If this activity is unexpected, your credentials may be compromised. For more information, see Remediating potentially compromised AWS credentials.

Stealth:IAMUser/PasswordPolicyChange

Account password policy was weakened.

Default severity: Low*

> **Note**
>
> This finding's severity can be Low, Medium, or High depending on the severity of the changes made to password policy.

- **Data source:** CloudTrail management events

The AWS account password policy was weakened on the listed account within your AWS environment. For example, it was deleted or updated to require fewer characters, not require symbols and numbers, or required to extend the password expiration period. This finding can also be triggered by an attempt to update or delete your AWS account password policy. The AWS account password policy defines the rules that govern what kinds of passwords can be set for your

IAM users. A weaker password policy permits the creation of passwords that are easy to remember and potentially easier to guess, thereby creating a security risk.

Remediation recommendations:

If this activity is unexpected, your credentials may be compromised. For more information, see Remediating potentially compromised AWS credentials.

UnauthorizedAccess:IAMUser/ConsoleLoginSuccess.B

Multiple worldwide successful console logins were observed.

Default severity: Medium

- **Data source:** CloudTrail management events

This finding informs you that multiple successful console logins for the same IAM user were observed around the same time in various geographical locations. Such anomalous and risky access location patterns indicate potential unauthorized access to your AWS resources.

Remediation recommendations:

If this activity is unexpected, your credentials may be compromised. For more information, see Remediating potentially compromised AWS credentials.

UnauthorizedAccess:IAMUser/InstanceCredentialExfiltration.InsideAWS

Credentials that were created exclusively for an EC2 instance through an Instance launch role are being used from another account within AWS.

Default severity: High*

> **Note**
>
> This finding's default severity is High. However, if the API was invoked by an account affiliated with your AWS environment, the severity is Medium.

- **Data source:** CloudTrail management events or CloudTrail data events for S3

This finding informs you when your Amazon EC2 instance credentials are used to invoke APIs from an IP address or an Amazon VPC endpoint, that is owned by a different AWS account than the one that the associated Amazon EC2 instance is running in. VPC endpoint detection is only available for services that support network activity events for VPC endpoints. For information about services that support network activity events for VPC endpoints, see Logging network activity events in the *AWS CloudTrail User Guide*.

AWS does not recommend redistributing temporary credentials outside of the entity that created them (for example, AWS applications, Amazon EC2, or AWS Lambda). However, authorized users can export credentials from their Amazon EC2 instances to make legitimate API calls. If the `remoteAccountDetails.Affiliated` field is `True` the API was invoked from an account associated with the same administrator account. To rule out a potential attack and verify the legitimacy of the activity, contact the AWS account owner or IAM principal to whom these credentials are assigned.

> **Note**
>
> If GuardDuty observes continued activity from a remote account, its machine learning (ML) model will identify this as an expected behavior. Therefore, GuardDuty will stop generating this finding for activity from that remote account. GuardDuty will continue to generate findings for new behavior from other remote accounts and will re-evaluate learned remote accounts as the behavior changes over time.

Remediation recommendations:

This finding gets generated when AWS API requests are made inside AWS through an Amazon EC2 instance outside of your AWS account, by using your Amazon EC2 instance's session credentials. It may be customary, such as for Transit Gateway architecture in a hub and spoke configuration, to route traffic through a single hub egress VPC with AWS service endpoints. If this behavior is expected, then GuardDuty recommends you to use Suppression rules and create a rule with a two-filter criteria. The first criteria is the finding type, which, in this case, is UnauthorizedAccess:IAMUser/InstanceCredentialExfiltration.InsideAWS. The second filter criteria is the remote account ID of the remote account details.

In response to this finding you can use the following workflow to determine a course of action:

1. Identify the remote account involved from the `service.action.awsApiCallAction.remoteAccountDetails.accountId` field.
2. Determine if that account is affiliated with your GuardDuty environment from the `service.action.awsApiCallAction.remoteAccountDetails.affiliated` field.
3. If the account **is** affiliated, contact the remote account owner and the owner of the Amazon EC2 instance credentials to investigate.

   If the account **is not** affiliated, then the first step is to evaluate if that account is associated with your organization but is not a part of your GuardDuty multiple-account environment set up, or if GuardDuty has not yet been enabled in this account. Next, contact the owner of the Amazon EC2 instance credentials to determine if there is a use case for a remote account to use these credentials.
4. If the owner of the credentials does not recognize the remote account the credentials may have been compromised by a threat actor operating within AWS. You should take the steps recommended in Remediating a potentially compromised Amazon EC2 instance, to secure your environment.

   Additionally, you can submit an abuse report to the AWS Trust and Safety team to begin an investigation into the remote account. When submitting your report to AWS Trust and Safety, include the full JSON details of the finding.

UnauthorizedAccess:IAMUser/InstanceCredentialExfiltration.OutsideAWS

Credentials that were created exclusively for an EC2 instance through an Instance launch role are being used from an external IP address.

Default severity: High

- **Data source:** CloudTrail management events or CloudTrail data events for S3

This finding informs you that a host outside of AWS has attempted to run AWS API operations using temporary AWS credentials that were created on an EC2 instance in your AWS environment. The listed EC2 instance might be compromised, and the temporary credentials from this instance might have been exfiltrated to a remote host outside of AWS. AWS does not recommend redistributing temporary credentials outside of the entity that created them (for example, AWS applications, EC2, or Lambda). However, authorized users can export credentials from their EC2 instances to make legitimate API calls. To rule out a potential attack and verify the legitimacy of the activity, validate if the use of instance credentials from the remote IP in the finding is expected.

> **Note**
>
> If GuardDuty observes continued activity from a remote account, its machine learning (ML) model will identify this as an expected behavior. Therefore, GuardDuty will stop generating this finding for activity from that remote account. GuardDuty will continue to generate findings for new behavior from other remote accounts and will re-evaluate learned remote accounts as the behavior changes over time.

Remediation recommendations:

This finding is generated when networking is configured to route internet traffic such that it egresses from an on-premises gateway rather than from a VPC Internet Gateway (IGW). Common configurations, such as using AWS Outposts, or VPC VPN connections, can result in traffic routed this way. If this is expected behavior, we recommend that you use suppression rules and create a rule that consists of two filter criteria. The first criteria is **finding type**, which should be `UnauthorizedAccess:IAMUser/InstanceCredentialExfiltration.OutsideAWS`. The second filter criteria is **API caller IPv4 Address** with the IP address or CIDR range of your on-premises internet gateway. To learn more about creating suppression rules see Suppression rules in GuardDuty.

> **Note**
>
> If GuardDuty observes continued activity from an external source its machine learning model will identify this as expected behavior and stop generating this finding for activity from that source. GuardDuty will continue to generate findings for new behavior from other sources, and will reevaluate learned sources as behavior changes over time.

If this activity is unexpected your credentials may be compromised, see Remediating potentially compromised AWS credentials.

UnauthorizedAccess:IAMUser/MaliciousIPCaller

An API was invoked from a known malicious IP address.

Default severity: Medium

- **Data source:** CloudTrail management events

This finding informs you that an API operation (for example, an attempt to launch an EC2 instance, create a new IAM user, or modify your AWS privileges) was invoked from a known malicious IP address. This can indicate unauthorized access to AWS resources within your environment.

Remediation recommendations:

If this activity is unexpected, your credentials may be compromised. For more information, see Remediating potentially compromised AWS credentials.

UnauthorizedAccess:IAMUser/MaliciousIPCaller.Custom

An API was invoked from an IP address on a custom threat list.

Default severity: Medium

- **Data source:** CloudTrail management events

This finding informs you that an API operation (for example, an attempt to launch an EC2 instance, create a new IAM user, or modify AWS privileges) was invoked from an IP address that is included on a threat list that you uploaded. In GuardDuty, a threat list consists of known malicious IP addresses. This can indicate unauthorized access to AWS resources within your environment.

Remediation recommendations:

If this activity is unexpected, your credentials may be compromised. For more information, see Remediating potentially compromised AWS credentials.

UnauthorizedAccess:IAMUser/TorIPCaller

An API was invoked from a Tor exit node IP address.

Default severity: Medium

- **Data source:** CloudTrail management events

This finding informs you that an API operation (for example, an attempt to launch an EC2 instance, create a new IAM user, or modify your AWS privileges) was invoked from a Tor exit node IP address. Tor is software for enabling anonymous communication. It encrypts and randomly bounces communications through relays between a series of network nodes. The last Tor node is called the exit node. This can indicate unauthorized access to your AWS resources with the intent of hiding the attacker's true identity.

Remediation recommendations:

If this activity is unexpected, your credentials may be compromised. For more information, see Remediating potentially compromised AWS credentials.

GuardDuty attack sequence finding types

GuardDuty detects an attack sequence when a specific sequence of multiple actions align to a potentially suspicious activity. An attack sequence includes **signals** such as API activities and GuardDuty findings. When GuardDuty observes a group of signals in a specific sequence that indicates an in-progress, ongoing, or a recent security threat, GuardDuty generates an attack sequence finding. GuardDuty considers individual API activities as weak signals because they don't present themselves as potential threat.

The attack sequence detections focus on potential compromise of Amazon S3 data (that can be a part of a broader ransomware attack), compromised AWS credentials, and compromised Amazon EKS clusters. The following sections provide details about each of the attack sequences.

Topics

- AttackSequence:EKS/CompromisedCluster
- AttackSequence:IAM/CompromisedCredentials
- AttackSequence:S3/CompromisedData

AttackSequence:EKS/CompromisedCluster

A sequence of suspicious actions performed by potentially compromised Amazon EKS cluster.

- Default severity: Critical
- Data sources:
  - EKS audit log events
  - Runtime Monitoring for Amazon EKS
  - Amazon EKS malware detection for Amazon EC2
  - AWS CloudTrail data events for S3
  - AWS CloudTrail management events
  - VPC Flow Logs
  - Route53 Resolver DNS query logs

This finding informs you that GuardDuty detected a sequence of suspicious actions that indicates a potentially compromised Amazon EKS cluster in your environment. Multiple suspicious and anomalous attack behaviors, such as malicious processes or connection with malicious endpoints, were observed in the same Amazon EKS cluster.

GuardDuty uses its proprietary correlation algorithms to observe and identify the sequence of actions performed by using the IAM credential. GuardDuty evaluates findings across protection plans and other signal sources to identify common and emerging attack patterns. GuardDuty uses multiple factors to surface threats, such as IP reputation, API sequences, user configuration, and potentially impacted resources.

Remediation actions: If this behavior is unexpected in your environment, then your Amazon EKS cluster may be compromised. For comprehensive remediation guidance, see Remediating EKS Protection findings and Remediating Runtime Monitoring findings.

Additionally, since AWS credentials may have been compromised through the EKS cluster, see Remediating potentially compromised AWS credentials. For steps to remediate other resources that may have been potentially impacted, see Remediating detected GuardDuty security findings.

AttackSequence:IAM/CompromisedCredentials

A sequence of API requests that were invoked by using potentially compromised AWS credentials.

- Default severity: Critical
- Data source: AWS CloudTrail management events

This finding informs you that GuardDuty detected a sequence of suspicious actions made by using AWS credentials that impacts one or more resources in your environment. Multiple suspicious and anomalous attack behaviors were observed by the same credentials, resulting in higher confidence that the credentials are being misused.

GuardDuty uses its proprietary correlation algorithms to observe and identify the sequence of actions performed by using the IAM credential. GuardDuty evaluates findings across protection plans and other signal sources to identify common and emerging attack patterns. GuardDuty uses multiple factors to surface threats, such as IP reputation, API sequences, user configuration, and potentially impacted resources.

Remediation actions: If this behavior is unexpected in your environment, then your AWS credentials may have been compromised. For steps to remediate, see Remediating potentially compromised AWS credentials. The compromised credentials may have been used to create or modify additional resources, such as Amazon S3 buckets, AWS Lambda functions, or Amazon EC2 instances, in your environment. For steps to remediate other resources that may have been potentially impacted, see Remediating detected GuardDuty security findings.

AttackSequence:S3/CompromisedData

A sequence of API requests was invoked in a potential attempt to exfiltrate or destroy data in Amazon S3.

- Default severity: Critical
- Data sources: AWS CloudTrail data events for S3 and AWS CloudTrail management events

This finding informs you that GuardDuty detected a sequence of suspicious actions indicative of data compromise in one or more Amazon Simple Storage Service (Amazon S3) buckets, by using potentially compromised AWS credentials. Multiple suspicious and anomalous attack behaviors (API requests) were observed, resulting in higher confidence of the credentials are being misused.

GuardDuty uses its correlation algorithms to observe and identify the sequence of actions performed by using the IAM credential. GuardDuty then evaluates findings across protection plans and other signal sources to identify common and emerging attack patterns. GuardDuty uses multiple factors to surface threats, such as IP reputation, API sequences, user configuration, and potentially impacted resources.

Remediation actions: If this activity is unexpected in your environment, your AWS credentials, or Amazon S3 data may have potentially exfiltrated or destroyed. For steps to remediate, see Remediating potentially compromised AWS credentials and Remediating a potentially compromised S3 bucket.

GuardDuty S3 Protection finding types

The following findings are specific to Amazon S3 resources and will have a **Resource Type** of `S3Bucket` if the data source is **CloudTrail data events for S3**, or `AccessKey` if the data source is **CloudTrail management events**. The severity and details of the findings will differ based on the finding type and the permission associated with the bucket.

The findings listed here include the data sources and models used to generate that finding type. For more information data sources and models, see GuardDuty foundational data sources.

> **Important**
>
> Findings with a data source of **CloudTrail data events for S3** are only generated if you have enabled S3 Protection. By default, after July 31, 2020, S3 Protection is enabled when an account enables GuardDuty for the first time, or when a delegated GuardDuty administrator account enables GuardDuty in an existing member account. However, when a new member joins the GuardDuty organization, the organization's auto-enable preferences will apply. For information about auto-enable preferences, see Setting organization auto-enable preferences. For information about how to enable S3 Protection, see GuardDuty S3 Protection

For all `S3Bucket` type findings, it is recommended that you examine the permissions on the bucket in question and the permissions of any users involved in the finding, if the activity is unexpected see the remediation recommendations detailed in Remediating a potentially compromised S3 bucket.

Topics

- Discovery:S3/AnomalousBehavior
- Discovery:S3/MaliciousIPCaller
- Discovery:S3/MaliciousIPCaller.Custom
- Discovery:S3/TorIPCaller
- Exfiltration:S3/AnomalousBehavior
- Exfiltration:S3/MaliciousIPCaller
- Impact:S3/AnomalousBehavior.Delete
- Impact:S3/AnomalousBehavior.Permission
- Impact:S3/AnomalousBehavior.Write
- Impact:S3/MaliciousIPCaller
- PenTest:S3/KaliLinux
- PenTest:S3/ParrotLinux
- PenTest:S3/PentooLinux
- Policy:S3/AccountBlockPublicAccessDisabled
- Policy:S3/BucketAnonymousAccessGranted
- Policy:S3/BucketBlockPublicAccessDisabled
- Policy:S3/BucketPublicAccessGranted
- Stealth:S3/ServerAccessLoggingDisabled
- UnauthorizedAccess:S3/MaliciousIPCaller.Custom
- UnauthorizedAccess:S3/TorIPCaller

Discovery:S3/AnomalousBehavior

An API commonly used to discover S3 objects was invoked in an anomalous way.

Default severity: Low

- **Data source:** CloudTrail data events for S3

This finding informs you that an IAM entity has invoked an S3 API to discover S3 buckets in your environment, such as `ListObjects`. This type of activity is associated with the discovery stage of an attack wherein an attacker gathers information to determine if your AWS environment is susceptible to a broader attack. This activity is suspicious because the IAM entity invoked the API in an unusual way. For example, an IAM entity with no previous history invokes an S3 API, or an IAM entity invokes an S3 API from an unusual location.

This API was identified as anomalous by GuardDuty's anomaly detection machine learning (ML) model. The ML model evaluates all the API requests in your account and identifies anomalous events that are associated with techniques used by adversaries. It tracks various factors of the API requests, such as the user who made the request, the location from which the request was made, the specific API that was requested, the bucket that was requested, and the number of API calls made. For more information on which factors of the API request are unusual for the user identity that invoked the request, see Finding details.

Remediation recommendations:

If this activity is unexpected for the associated principal, it may indicate that the credentials have been exposed or your S3 permissions are not restrictive enough. For more information, see Remediating a potentially compromised S3 bucket.

Discovery:S3/MaliciousIPCaller

An S3 API commonly used to discover resources in an AWS environment was invoked from a known malicious IP address.

Default severity: High

- **Data source:** CloudTrail data events for S3

This finding informs you that an S3 API operation was invoked from an IP address that is associated with known malicious activity. The observed API is commonly associated with the discovery stage of an attack when an adversary is gathering information about your AWS environment. Examples include `GetObjectAcl` and `ListObjects`.

Remediation recommendations:

If this activity is unexpected for the associated principal, it may indicate that the credentials have been exposed or your S3 permissions are not restrictive enough. For more information, see Remediating a potentially compromised S3 bucket.

Discovery:S3/MaliciousIPCaller.Custom

An S3 API was invoked from an IP address on a custom threat list.

Default severity: High

- **Data source:** CloudTrail data events for S3

This finding informs you that an S3 API, such as `GetObjectAcl` or `ListObjects`, was invoked from an IP address that is included on a threat list that you uploaded. The threat list associated with this finding is listed in the **Additional information** section of a finding's details. This type of activity is associated with the discovery stage of an attack wherein an attacker is gathering information to determine if your AWS environment is susceptible to a broader attack.

Remediation recommendations:

If this activity is unexpected for the associated principal, it may indicate that the credentials have been exposed or your S3 permissions are not restrictive enough. For more information, see Remediating a potentially compromised S3 bucket.

Discovery:S3/TorIPCaller

An S3 API was invoked from a Tor exit node IP address.

Default severity: Medium

- **Data source:** CloudTrail data events for S3

This finding informs you that an S3 API, such as `GetObjectAcl` or `ListObjects`, was invoked from a Tor exit node IP address. This type of activity is associated with the discovery stage of an attack wherein an attacker is gathering information to determine if your AWS environment is susceptible to a broader attack. Tor is software for enabling anonymous communication. It encrypts and randomly bounces communications through relays between a series of network nodes. The last Tor node is called the exit node. This can indicate unauthorized access to your AWS resources with the intent of hiding the attacker's true identity.

Remediation recommendations:

If this activity is unexpected for the associated principal, it may indicate that the credentials have been exposed or your S3 permissions are not restrictive enough. For more information, see Remediating a potentially compromised S3 bucket.

Exfiltration:S3/AnomalousBehavior

An IAM entity invoked an S3 API in a suspicious way.

Default severity: High

- **Data source:** CloudTrail data events for S3

This finding informs you that an IAM entity is making API calls that involve an S3 bucket and this activity differs from that entity's established baseline. The API call used in this activity is associated with the exfiltration stage of an attack, wherein an attacker attempts to collect data. This activity is suspicious because the IAM entity invoked the API in an unusual way. For example, an IAM entity with no previous history invokes an S3 API, or an IAM entity invokes an S3 API from an unusual location.

This API was identified as anomalous by GuardDuty's anomaly detection machine learning (ML) model. The ML model evaluates all the API requests in your account and identifies anomalous events that are associated with techniques used by adversaries. It tracks various factors of the API requests, such as the user who made the request, the location from which the request was made, the specific API that was requested, the bucket that was requested, and the number of API calls made. For more information on which factors of the API request are unusual for the user identity that invoked the request, see Finding details.

Remediation recommendations:

If this activity is unexpected for the associated principal, it may indicate that the credentials have been exposed or your S3 permissions are not restrictive enough. For more information, see Remediating a potentially compromised S3 bucket.

Exfiltration:S3/MaliciousIPCaller

An S3 API commonly used to collect data from an AWS environment was invoked from a known malicious IP address.

Default severity: High

- **Data source:** CloudTrail data events for S3

This finding informs you that an S3 API operation was invoked from an IP address that is associated with known malicious activity. The observed API is commonly associated with exfiltration tactics where an adversary is trying to collect data from your network. Examples include `GetObject` and `CopyObject`.

Remediation recommendations:

If this activity is unexpected for the associated principal, it may indicate that the credentials have been exposed or your S3 permissions are not restrictive enough. For more information, see Remediating a potentially compromised S3 bucket.

Impact:S3/AnomalousBehavior.Delete

An IAM entity invoked an S3 API that attempts to delete data in a suspicious way.

Default severity: High

- **Data source:** CloudTrail data events for S3

This finding informs you that an IAM entity in your AWS environment is making API calls that involve an S3 bucket, and this behavior differs from that entity's established baseline. The API call used in this activity is associated with an attack that attempts to delete data. This activity is suspicious because the IAM entity invoked the API in an unusual way. For example, an IAM entity with no previous history invokes an S3 API, or an IAM entity invokes an S3 API from an unusual location.

This API was identified as anomalous by GuardDuty's anomaly detection machine learning (ML) model. The ML model evaluates all the API requests in your account and identifies anomalous events that are associated with techniques used by adversaries. It tracks various factors of the API requests, such as the user who made the request, the location from which the request was made, the specific API that was requested, the bucket that was requested, and the number of API calls made. For more information on which factors of the API request are unusual for the user identity that invoked the request, see Finding details.

Remediation recommendations:

If this activity is unexpected for the associated principal, it may indicate that the credentials have been exposed or your S3 permissions are not restrictive enough. For more information, see Remediating a potentially compromised S3 bucket.

We recommend an audit of your S3 bucket's contents to determine if you the previous object version can or should be restored.

Impact:S3/AnomalousBehavior.Permission

An API commonly used to set the access control list (ACL) permissions was invoked in an anomalous way.

Default severity: High

- **Data source:** CloudTrail data events for S3

This finding informs you that an IAM entity in your AWS environment has changed a bucket policy or ACL on the listed S3 buckets. This change may publicly expose your S3 buckets to all the authenticated AWS users.

This API was identified as anomalous by GuardDuty's anomaly detection machine learning (ML) model. The ML model evaluates all the API requests in your account and identifies anomalous events that are associated with techniques used by adversaries. It tracks various factors of the API requests, such as the user who made the request, the location from which the request was made, the specific API that was requested, the bucket that was requested, and the number of API calls made. For more information on which factors of the API request are unusual for the user identity that invoked the request, see Finding details.

Remediation recommendations:

If this activity is unexpected for the associated principal, it may indicate that the credentials have been exposed or your S3 permissions are not restrictive enough. For more information, see Remediating a potentially compromised S3 bucket.

We recommend an audit of your S3 bucket's contents to ensure that no objects were unexpectedly allowed to be accessed publicly.

Impact:S3/AnomalousBehavior.Write

An IAM entity invoked an S3 API that attempts to write data in a suspicious way.

Default severity: Medium

- **Data source:** CloudTrail data events for S3

This finding informs you that an IAM entity in your AWS environment is making API calls that involve an S3 bucket, and this behavior differs from that entity's established baseline. The API call used in this activity is associated with an attack that attempts to write data. This activity is suspicious because the IAM entity invoked the API in an unusual way. For example, an IAM entity with no previous history invokes an S3 API, or an IAM entity invokes an S3 API from an unusual location.

This API was identified as anomalous by GuardDuty's anomaly detection machine learning (ML) model. The ML model evaluates all the API requests in your account and identifies anomalous events that are associated with techniques used by adversaries. It tracks various factors of the API requests, such as the user who made the request, the location from which the request was made, the specific API that was requested, the bucket that was requested, and the number of API calls made. For more information on which factors of the API request are unusual for the user identity that invoked the request, see Finding details.

Remediation recommendations:

If this activity is unexpected for the associated principal, it may indicate that the credentials have been exposed or your S3 permissions are not restrictive enough. For more information, see Remediating a potentially compromised S3 bucket.

We recommend an audit of your S3 bucket's contents to ensure that this API call didn't write malicious or unauthorized data.

Impact:S3/MaliciousIPCaller

An S3 API commonly used to tamper with data or processes in an AWS environment was invoked from a known malicious IP address.

Default severity: High

- **Data source:** CloudTrail data events for S3

This finding informs you that an S3 API operation was invoked from an IP address that is associated with known malicious activity. The observed API is commonly associated with impact tactics where an adversary is trying manipulate, interrupt, or destroy data within your AWS environment. Examples include `PutObject` and `PutObjectAcl`.

Remediation recommendations:

If this activity is unexpected for the associated principal, it may indicate that the credentials have been exposed or your S3 permissions are not restrictive enough. For more information, see Remediating a potentially compromised S3 bucket.

PenTest:S3/KaliLinux

An S3 API was invoked from a Kali Linux machine.

Default severity: Medium

- **Data source:** CloudTrail data events for S3

This finding informs you that a machine running Kali Linux is making S3 API calls using credentials that belong to your AWS account. Your credentials might be compromised. Kali Linux is a popular penetration testing tool that security professionals use to identify weaknesses in EC2 instances that require patching. Attackers also use this tool to find EC2 configuration weaknesses and gain unauthorized access to your AWS environment.

Remediation recommendations:

If this activity is unexpected for the associated principal, it may indicate that the credentials have been exposed or your S3 permissions are not restrictive enough. For more information, see Remediating a potentially compromised S3 bucket.

PenTest:S3/ParrotLinux

An S3 API was invoked from a Parrot Security Linux machine.

Default severity: Medium

- **Data source:** CloudTrail data events for S3

This finding informs you that a machine running Parrot Security Linux is making S3 API calls using credentials that belong to your AWS account. Your credentials might be compromised. Parrot Security Linux is a popular penetration testing tool that security professionals use to identify weaknesses in EC2 instances that require patching. Attackers also use this tool to find EC2 configuration weaknesses and gain unauthorized access to your AWS environment.

Remediation recommendations:

If this activity is unexpected for the associated principal, it may indicate that the credentials have been exposed or your S3 permissions are not restrictive enough. For more information, see Remediating a potentially compromised S3 bucket.

PenTest:S3/PentooLinux

An S3 API was invoked from a Pentoo Linux machine.

Default severity: Medium

- **Data source:** CloudTrail data events for S3

This finding informs you that a machine running Pentoo Linux is making S3 API calls using credentials that belong to your AWS account. Your credentials might be compromised. Pentoo Linux is a popular penetration testing tool that security professionals use to identify weaknesses in EC2 instances that require patching. Attackers also use this tool to find EC2 configuration weaknesses and gain unauthorized access to your AWS environment.

Remediation recommendations:

If this activity is unexpected for the associated principal, it may indicate that the credentials have been exposed or your S3 permissions are not restrictive enough. For more information, see Remediating a potentially compromised S3 bucket.

Policy:S3/AccountBlockPublicAccessDisabled

An IAM entity invoked an API used to disable S3 Block Public Access on an account.

Default severity: Low

- **Data source:** CloudTrail management events

This finding informs you that Amazon S3 Block Public Access was disabled at the account level. When S3 Block Public Access settings are enabled, they are used to filter the policies or access control lists (ACLs) on buckets as a security measure to prevent inadvertent public exposure of data.

Typically, S3 Block Public Access is turned off in an account to allow public access to a bucket or to the objects in the bucket. When S3 Block Public Access is disabled for an account, access to your buckets is controlled by the policies, ACLs, or bucket-level Block Public Access settings applied to your individual buckets. This does not necessarily mean that the buckets are shared publicly, but that you should audit the permissions applied to the buckets to confirm that they provide the appropriate level of access.

Remediation recommendations:

If this activity is unexpected for the associated principal, it may indicate that the credentials have been exposed or your S3 permissions are not restrictive enough. For more information, see Remediating a potentially compromised S3 bucket.

Policy:S3/BucketAnonymousAccessGranted

An IAM principal has granted access to an S3 bucket to the internet by changing bucket policies or ACLs.

Default severity: High

- **Data source:** CloudTrail management events

This finding informs you that the listed S3 bucket has been made publicly accessible on the internet because an IAM entity has changed a bucket policy or ACL on that bucket.

After a policy or ACL change is detected, GuardDuty uses automated reasoning powered by Zelkova, to determine if the bucket is publicly accessible.

> **Note**
>
> If a bucket's ACLs or bucket policies are configured to explicitly deny or to deny all, this finding may not reflect the current state of the bucket. This finding will not reflect any S3 Block Public Access settings that may have been enabled for your S3 bucket. In such cases, the `effectivePermission` value in the finding will be marked as `UNKNOWN`.

Remediation recommendations:

If this activity is unexpected for the associated principal, it may indicate that the credentials have been exposed or your S3 permissions are not restrictive enough. For more information, see Remediating a potentially compromised S3 bucket.

Policy:S3/BucketBlockPublicAccessDisabled

An IAM entity invoked an API used to disable S3 Block Public Access on a bucket.

Default severity: Low

- **Data source:** CloudTrail management events

This finding informs you that Block Public Access was disabled for the listed S3 bucket. When enabled, S3 Block Public Access settings are used to filter the policies or access control lists (ACLs) applied to buckets as a security measure to prevent inadvertent public exposure of data.

Typically, S3 Block Public Access is turned off on a bucket to allow public access to the bucket or to the objects within. When S3 Block Public Access is disabled for a bucket, access to the bucket is controlled by the policies or ACLs applied to it. This does not mean that the bucket is shared publicly, but you should audit the policies and ACLs applied to the bucket to confirm that appropriate permissions are applied.

Remediation recommendations:

If this activity is unexpected for the associated principal, it may indicate that the credentials have been exposed or your S3 permissions are not restrictive enough. For more information, see Remediating a potentially compromised S3 bucket.

Policy:S3/BucketPublicAccessGranted

An IAM principal has granted public access to an S3 bucket to all AWS users by changing bucket policies or ACLs.

Default severity: High

- **Data source:** CloudTrail management events

This finding informs you that the listed S3 bucket has been publicly exposed to all authenticated AWS users because an IAM entity has changed a bucket policy or ACL on that S3 bucket.

After a policy or ACL change is detected, GuardDuty uses automated reasoning powered by Zelkova, to determine if the bucket is publicly accessible.

> **Note**
>
> If a bucket's ACLs or bucket policies are configured to explicitly deny or to deny all, this finding may not reflect the current state of the bucket. This finding will not reflect any S3 Block Public Access settings that may have been enabled for your S3 bucket. In such cases, the `effectivePermission` value in the finding will be marked as `UNKNOWN`.

Remediation recommendations:

If this activity is unexpected for the associated principal, it may indicate that the credentials have been exposed or your S3 permissions are not restrictive enough. For more information, see Remediating a potentially compromised S3 bucket.

Stealth:S3/ServerAccessLoggingDisabled

S3 server access logging was disabled for a bucket.

Default severity: Low

- **Data source:** CloudTrail management events

This finding informs you that S3 server access logging is disabled for a bucket within your AWS environment. If disabled, no web request logs are created for any attempts to access the identified S3 bucket, however, S3 management API calls to the bucket, such as DeleteBucket, are still tracked. If S3 data event logging is enabled through CloudTrail for this bucket, web requests for objects within the bucket will still be tracked. Disabling logging is a technique used by unauthorized users in order to evade detection. To learn more about S3 logs, see S3 Server Access Logging and S3 Logging Options .

Remediation recommendations:

If this activity is unexpected for the associated principal, it may indicate that the credentials have been exposed or your S3 permissions are not restrictive enough. For more information, see Remediating a potentially compromised S3 bucket.

UnauthorizedAccess:S3/MaliciousIPCaller.Custom

An S3 API was invoked from an IP address on a custom threat list.

Default severity: High

- **Data source:** CloudTrail data events for S3

This finding informs you that an S3 API operation, for example, `PutObject` or `PutObjectAcl`, was invoked from an IP address that is included on a threat list that you uploaded. The threat list associated with this finding is listed in the **Additional information** section of a finding's details.

Remediation recommendations:

If this activity is unexpected for the associated principal, it may indicate that the credentials have been exposed or your S3 permissions are not restrictive enough. For more information, see Remediating a potentially compromised S3 bucket.

UnauthorizedAccess:S3/TorIPCaller

An S3 API was invoked from a Tor exit node IP address.

Default severity: High

- **Data source:** CloudTrail data events for S3

This finding informs you that an S3 API operation, such as `PutObject` or `PutObjectAcl`, was invoked from a Tor exit node IP address. Tor is software for enabling anonymous communication. It encrypts and randomly bounces communications through relays between a series of network nodes. The last Tor node is called the exit node. This finding can indicate unauthorized access to your AWS resources with the intent of hiding the attacker's true identity.

Remediation recommendations:

If this activity is unexpected for the associated principal, it may indicate that the credentials have been exposed or your S3 permissions are not restrictive enough. For more information, see Remediating a potentially compromised S3 bucket.

EKS Protection finding types

The following findings are specific to Amazon EKS resources and have a **resource_type** of `EKSCluster`. The severity and details of the findings differ based on finding type.

For all EKS audit logs type findings we recommend that you examine the resource in question to determine if the activity is expected or potentially malicious. For guidance on remediating a compromised EKS audit logs resource identified by a GuardDuty finding, see Remediating EKS Protection findings.

> **Note**
>
> If the activity because of which these findings get generated is expected, consider adding Suppression rules in GuardDuty to prevent future alerts.

Topics

- CredentialAccess:Kubernetes/MaliciousIPCaller

- CredentialAccess:Kubernetes/MaliciousIPCaller.Custom
- CredentialAccess:Kubernetes/SuccessfulAnonymousAccess
- CredentialAccess:Kubernetes/TorIPCaller
- DefenseEvasion:Kubernetes/MaliciousIPCaller
- DefenseEvasion:Kubernetes/MaliciousIPCaller.Custom
- DefenseEvasion:Kubernetes/SuccessfulAnonymousAccess
- DefenseEvasion:Kubernetes/TorIPCaller
- Discovery:Kubernetes/MaliciousIPCaller
- Discovery:Kubernetes/MaliciousIPCaller.Custom
- Discovery:Kubernetes/SuccessfulAnonymousAccess
- Discovery:Kubernetes/TorIPCaller
- Execution:Kubernetes/ExecInKubeSystemPod
- Impact:Kubernetes/MaliciousIPCaller
- Impact:Kubernetes/MaliciousIPCaller.Custom
- Impact:Kubernetes/SuccessfulAnonymousAccess
- Impact:Kubernetes/TorIPCaller
- Persistence:Kubernetes/ContainerWithSensitiveMount
- Persistence:Kubernetes/MaliciousIPCaller
- Persistence:Kubernetes/MaliciousIPCaller.Custom
- Persistence:Kubernetes/SuccessfulAnonymousAccess
- Persistence:Kubernetes/TorIPCaller
- Policy:Kubernetes/AdminAccessToDefaultServiceAccount
- Policy:Kubernetes/AnonymousAccessGranted
- Policy:Kubernetes/ExposedDashboard
- Policy:Kubernetes/KubeflowDashboardExposed
- PrivilegeEscalation:Kubernetes/PrivilegedContainer
- CredentialAccess:Kubernetes/AnomalousBehavior.SecretsAccessed
- PrivilegeEscalation:Kubernetes/AnomalousBehavior.RoleBindingCreated
- Execution:Kubernetes/AnomalousBehavior.ExecInPod
- PrivilegeEscalation:Kubernetes/AnomalousBehavior.WorkloadDeployed!PrivilegedContainer

- Persistence:Kubernetes/AnomalousBehavior.WorkloadDeployed!ContainerWithSensitiveMount
- Execution:Kubernetes/AnomalousBehavior.WorkloadDeployed
- PrivilegeEscalation:Kubernetes/AnomalousBehavior.RoleCreated
- Discovery:Kubernetes/AnomalousBehavior.PermissionChecked

> **Note**
>
> Before Kubernetes version 1.14, the `system:unauthenticated` group was associated to `system:discovery` and `system:basic-user` **ClusterRoles** by default. This association may allow unintended access from anonymous users. Cluster updates do not revoke these permissions. Even if you updated your cluster to version 1.14 or higher, these permissions may still be enabled. We recommend that you disassociate these permissions from the `system:unauthenticated` group. For guidance on revoking these permissions, see Security best practices for Amazon EKS in the *Amazon EKS User Guide*.

CredentialAccess:Kubernetes/MaliciousIPCaller

An API commonly used to access credentials or secrets in a Kubernetes cluster was invoked from a known malicious IP address.

Default severity: High

- **Feature:** EKS audit logs

This finding informs you that an API operation was invoked from an IP address that is associated with known malicious activity. The API observed is commonly associated with the credential access tactics where an adversary is attempting to collect passwords, usernames, and access keys for your Kubernetes cluster.

Remediation recommendations:

If the user reported in the finding under the `KubernetesUserDetails` section is `system:anonymous`, investigate why the anonymous user was permitted to invoke the API and revoke the permissions, if needed, by following the instructions in Security best practices for Amazon EKS in the *Amazon EKS User Guide*. If the user is an authenticated user, investigate to determine if the activity was legitimate or malicious. If the activity was malicious revoke access of the user and reverse any changes made by an adversary to your cluster. For more information, see Remediating EKS Protection findings.

CredentialAccess:Kubernetes/MaliciousIPCaller.Custom

An API commonly used to access credentials or secrets in a Kubernetes cluster was invoked from an IP address on a custom threat list.

Default severity: High

- **Feature:** EKS audit logs

This finding informs you that an API operation was invoked from an IP address that is included on a threat list that you uploaded. The threat list associated with this finding is listed in the **Additional Information** section of a finding's details. The API observed is commonly associated with the credential access tactics where an adversary is attempting to collect passwords, usernames, and access keys for your Kubernetes cluster.

Remediation recommendations:

If the user reported in the finding under the `KubernetesUserDetails` section is `system:anonymous`, investigate why the anonymous user was permitted to invoke the API and and revoke the permissions, if needed, by following the instructions in Security best practices for Amazon EKS in the *Amazon EKS User Guide*. If the user is an authenticated user, investigate to determine if the activity was legitimate or malicious. If the activity was malicious revoke access of the user and reverse any changes made by an adversary to your cluster. For more information, see Remediating EKS Protection findings.

CredentialAccess:Kubernetes/SuccessfulAnonymousAccess

An API commonly used to access credentials or secrets in a Kubernetes cluster was invoked by an unauthenticated user.

Default severity: High

- **Feature:** EKS audit logs

This finding informs you that an API operation was successfully invoked by the `system:anonymous` user. API calls made by `system:anonymous` are unauthenticated. The observed API is commonly associated with the credential access tactics where an adversary is attempting to collect passwords, usernames, and access keys for your Kubernetes cluster. This activity indicates that anonymous or unauthenticated access is permitted on the API action reported in the finding and may be permitted on other actions. If this behavior is not expected, it may indicate a configuration mistake or that your credentials are compromised.

Remediation recommendations:

You should examine the permissions that have been granted to the `system:anonymous` user on your cluster and ensure that all the permissions are needed. If the permissions were granted mistakenly or maliciously, you should revoke access of the user and reverse any changes made by an adversary to your cluster. For more information, see Security best practices for Amazon EKS in the *Amazon EKS User Guide*.

For more information, see Remediating EKS Protection findings.

CredentialAccess:Kubernetes/TorIPCaller

An API commonly used to access credentials or secrets in a Kubernetes cluster was invoked from a Tor exit node IP address.

Default severity: High

- **Feature:** EKS audit logs

This finding informs you that an API was invoked from a Tor exit node IP address. The API observed is commonly associated with the credential access tactics where an adversary is attempting to collect passwords, usernames, and access keys for your Kubernetes cluster. Tor is software for enabling anonymous communication. It encrypts and randomly bounces communications through relays between a series of network nodes. The last Tor node is called the exit node. This can indicate unauthorized access to your Kubernetes cluster resources with the intent of hiding the attacker's true identity.

Remediation recommendations:

If the user reported in the finding under the `KubernetesUserDetails` section is `system:anonymous`, investigate why the anonymous user was permitted to invoke the API and revoke the permissions, if needed, by following the instructions in [Security best practices for Amazon EKS] in the *Amazon EKS User Guide*. If the user is an authenticated user, investigate to determine if the activity was legitimate or malicious. If the activity was malicious revoke access of the user and reverse any changes made by an adversary to your cluster. For more information, see [Remediating EKS Protection findings].

DefenseEvasion:Kubernetes/MaliciousIPCaller

An API commonly used to evade defensive measures was invoked from a known malicious IP address.

Default severity: High

- **Feature:** EKS audit logs

This finding informs you that an API operation was invoked from an IP address that is associated with known malicious activity. The API observed is commonly associated with defense evasion tactics where an adversary is trying to hide their actions to avoid detection.

Remediation recommendations:

If the user reported in the finding under the `KubernetesUserDetails` section is `system:anonymous`, investigate why the anonymous user was permitted to invoke the API and revoke the permissions, if needed, by following the instructions in [Security best practices for Amazon EKS] in the *Amazon EKS User Guide*. If the user is an authenticated user, investigate to determine if the activity was legitimate or malicious. If the activity was malicious revoke access of the user and reverse any changes made by an adversary to your cluster. For more information, see [Remediating EKS Protection findings].

DefenseEvasion:Kubernetes/MaliciousIPCaller.Custom

An API commonly used to evade defensive measures was invoked from an IP address on a custom threat list.

Default severity: High

- **Feature:** EKS audit logs

This finding informs you that an API operation was invoked from an IP address that is included on a threat list that you uploaded. The threat list associated with this finding is listed in the **Additional Information** section of a finding's details. The API observed is commonly associated with defense evasion tactics where an adversary is trying to hide their actions to avoid detection.

Remediation recommendations:

If the user reported in the finding under the `KubernetesUserDetails` section is `system:anonymous`, investigate why the anonymous user was permitted to invoke the API and revoke the permissions, if needed, by following the instructions in [Security best practices for Amazon EKS] in the *Amazon EKS User Guide*. If the user is an authenticated user, investigate to determine if the activity was legitimate or malicious. If the activity was malicious revoke access of the user and reverse any changes made by an adversary to your cluster. For more information, see Remediating EKS Protection findings.

DefenseEvasion:Kubernetes/SuccessfulAnonymousAccess

An API commonly used to evade defensive measures was invoked by an unauthenticated user.

Default severity: High

- **Feature:** EKS audit logs

This finding informs you that an API operation was successfully invoked by the `system:anonymous` user. API calls made by `system:anonymous` are unauthenticated. The observed API is commonly associated with defense evasion tactics where an adversary is trying to hide their actions to avoid detection. This activity indicates that anonymous or unauthenticated access is permitted on the API action reported in the finding and may be permitted on other actions. If this behavior is not expected, it may indicate a configuration mistake or that your credentials are compromised.

Remediation recommendations:

You should examine the permissions that have been granted to the `system:anonymous` user on your cluster and ensure that all the permissions are needed. If the permissions were granted mistakenly or maliciously, you should revoke access of the user and reverse any changes made by an adversary to your cluster. For more information, see Security best practices for Amazon EKS in the *Amazon EKS User Guide*.

For more information, see Remediating EKS Protection findings.

DefenseEvasion:Kubernetes/TorIPCaller

An API commonly used to evade defensive measures was invoked from a Tor exit node IP address.

Default severity: High

- **Feature:** EKS audit logs

This finding informs you that an API was invoked from a Tor exit node IP address. The API observed is commonly associated with defense evasion tactics where an adversary is trying to hide their actions to avoid detection. Tor is software for enabling anonymous communication. It encrypts and randomly bounces communications through relays between a series of network nodes. The last Tor node is called the exit node. This can indicate unauthorized access to your Kubernetes cluster with the intent of hiding the adversary's true identity.

Remediation recommendations:

If the user reported in the finding under the `KubernetesUserDetails` section is `system:anonymous`, investigate why the anonymous user was permitted to invoke the API and revoke the permissions, if needed, by following the instructions in Security best practices for Amazon EKS in the *Amazon EKS User Guide*. If the user is an authenticated user, investigate to determine if the activity was legitimate or malicious. If the activity was malicious revoke access of the user and reverse any changes made by an adversary to your cluster. For more information, see Remediating EKS Protection findings.

Discovery:Kubernetes/MaliciousIPCaller

An API commonly used to discover resources in a Kubernetes cluster was invoked from an IP address.

Default severity: Medium

- **Feature:** EKS audit logs

This finding informs you that an API operation was invoked from an IP address that is associated with known malicious activity. The observed API is commonly used with the discovery stage of an attack wherein an attacker is gathering information to determine if your Kubernetes cluster is susceptible to a broader attack.

> **For unauthenticated access**
>
> MaliciousIPCaller findings are not generated for unauthenticated access. SuccessfulAnonymousAccess findings are generated for unauthenticated or anonymous access.

Remediation recommendations:

If the user reported in the finding under the `KubernetesUserDetails` section is `system:anonymous`, investigate why the anonymous user was permitted to invoke the API and revoke the permissions, if needed, by following the instructions in Security best practices for Amazon EKS in the *Amazon EKS User Guide*. If the user is an authenticated user, investigate to determine if the activity was legitimate or malicious. If the activity was malicious revoke access of the user and reverse any changes made by an adversary to your cluster. For more information, see Remediating EKS Protection findings.

Discovery:Kubernetes/MaliciousIPCaller.Custom

An API commonly used to discover resources in a Kubernetes cluster was invoked from an IP address on a custom threat list.

Default severity: Medium

- **Feature:** EKS audit logs

This finding informs you that an API was invoked from an IP address that is included on a threat list that you uploaded. The threat list associated with this finding is listed in the **Additional**

Information section of a finding's details. The observed API is commonly used with the discovery stage of an attack wherein an attacker is gathering information to determine if your Kubernetes cluster is susceptible to a broader attack.

Remediation recommendations:

If the user reported in the finding under the `KubernetesUserDetails` section is `system:anonymous`, investigate why the anonymous user was permitted to invoke the API and revoke the permissions, if needed, by following the instructions in [Security best practices for Amazon EKS] in the *Amazon EKS User Guide*. If the user is an authenticated user, investigate to determine if the activity was legitimate or malicious. If the activity was malicious revoke access of the user and reverse any changes made by an adversary to your cluster. For more information, see [Remediating EKS Protection findings].

Discovery:Kubernetes/SuccessfulAnonymousAccess

An API commonly used to discover resources in a Kubernetes cluster was invoked by an unauthenticated user.

Default severity: Medium

- **Feature:** EKS audit logs

This finding informs you that an API operation was successfully invoked by the `system:anonymous` user. API calls made by `system:anonymous` are unauthenticated. The observed API is commonly associated with the discovery stage of an attack when an adversary is gathering information on your Kubernetes cluster. This activity indicates that anonymous or unauthenticated access is permitted on the API action reported in the finding and may be permitted on other actions. If this behavior is not expected, it may indicate a configuration mistake or that your credentials are compromised.

This finding type excludes the health check API endpoints such as `/healthz`, `/livez`, `/readyz`, and `/version`.

Remediation recommendations:

You should examine the permissions that have been granted to the `system:anonymous` user on your cluster and ensure that all the permissions are needed. If the permissions were granted mistakenly or maliciously, you should revoke access of the user and reverse any changes made by an adversary to your cluster. For more information, see Security best practices for Amazon EKS in the *Amazon EKS User Guide*.

For more information, see Remediating EKS Protection findings.

Discovery:Kubernetes/TorIPCaller

An API commonly used to discover resources in a Kubernetes cluster was invoked from a Tor exit node IP address.

Default severity: Medium

- **Feature:** EKS audit logs

This finding informs you that an API was invoked from a Tor exit node IP address. The observed API is commonly used with the discovery stage of an attack wherein an attacker is gathering information to determine if your Kubernetes cluster is susceptible to a broader attack. Tor is software for enabling anonymous communication. It encrypts and randomly bounces communications through relays between a series of network nodes. The last Tor node is called the exit node. This can indicate unauthorized access to your Kubernetes cluster with the intent of hiding the adversary's true identity.

Remediation recommendations:

If the user reported in the finding under the `KubernetesUserDetails` section is `system:anonymous`, investigate why the anonymous user was permitted to invoke the APIand revoke the permissions, if needed, by following the instructions in Security best practices for Amazon EKS in the *Amazon EKS User Guide*. If the user is an authenticated user, investigate to determine if the activity was legitimate or malicious. If the activity was malicious revoke access of the user and reverse any changes made by an adversary to your cluster. For more information, see Remediating EKS Protection findings.

Execution:Kubernetes/ExecInKubeSystemPod

A command was executed inside a pod within the `kube-system` namespace

Default severity: Medium

- **Feature:** EKS audit logs

This finding informs you that a command was executed in a pod within the `kube-system` namespace using **Kubernetes exec API**. `kube-system` namespace is a default namespaces, which is primarily used for system level components such as `kube-dns` and `kube-proxy`. It is very uncommon to execute commands inside pods or containers under `kube-system` namespace and may indicate suspicious activity.

Remediation recommendations:

If the execution of this command is unexpected, the credentials of the user identity used to execute the command may be compromised. Revoke access of the user and reverse any changes made by an adversary to your cluster. For more information, see Remediating EKS Protection findings.

Impact:Kubernetes/MaliciousIPCaller

An API commonly used to tamper with resources in a Kubernetes cluster was invoked from a known malicious IP address.

Default severity: High

- **Feature:** EKS audit logs

This finding informs you that an API operation was invoked from an IP address that is associated with known malicious activity. The observed API is commonly associated with impact tactics where an adversary is trying to manipulate, interrupt, or destroy data within your AWS environment.

Remediation recommendations:

If the user reported in the finding under the `KubernetesUserDetails` section is `system:anonymous`, investigate why the anonymous user was permitted to invoke the API and revoke the permissions, if needed, by following the instructions in Security best practices for Amazon EKS in the *Amazon EKS User Guide*. If the user is an authenticated user, investigate to determine if the activity was legitimate or malicious. If the activity was malicious revoke access of the user and reverse any changes made by an adversary to your cluster. For more information, see Remediating EKS Protection findings.

Impact:Kubernetes/MaliciousIPCaller.Custom

An API commonly used to tamper with resources in a Kubernetes cluster was invoked from an IP address on a custom threat list.

Default severity: High

- **Feature:** EKS audit logs

This finding informs you that an API operation was invoked from an IP address that is included on a threat list that you uploaded. The threat list associated with this finding is listed in the **Additional Information** section of a finding's details. The observed API is commonly associated with impact tactics where an adversary is trying to manipulate, interrupt, or destroy data within your AWS environment.

Remediation recommendations:

If the user reported in the finding under the `KubernetesUserDetails` section is `system:anonymous`, investigate why the anonymous user was permitted to invoke the API and revoke the permissions, if needed, by following the instructions in [Security best practices for Amazon EKS] in the *Amazon EKS User Guide*. If the user is an authenticated user, investigate to determine if the activity was legitimate or malicious. If the activity was malicious revoke access of the user and reverse any changes made by an adversary to your cluster. For more information, see [Remediating EKS Protection findings].

Impact:Kubernetes/SuccessfulAnonymousAccess

An API commonly used to tamper with resources in a Kubernetes cluster was invoked by an unauthenticated user.

Default severity: High

- **Feature:** EKS audit logs

This finding informs you that an API operation was successfully invoked by the `system:anonymous` user. API calls made by `system:anonymous` are unauthenticated. The observed API is commonly associated with the impact stage of an attack when an adversary is tampering with resources in your cluster. This activity indicates that anonymous or unauthenticated access is permitted on the API action reported in the finding and may be permitted on other actions. If this behavior is not expected, it may indicate a configuration mistake or that your credentials are compromised.

Remediation recommendations:

You should examine the permissions that have been granted to the `system:anonymous` user on your cluster and ensure that all the permissions are needed. If the permissions were granted mistakenly or maliciously, you should revoke access of the user and reverse any changes made by an adversary to your cluster. For more information, see Security best practices for Amazon EKS in the *Amazon EKS User Guide*.

For more information, see Remediating EKS Protection findings.

Impact:Kubernetes/TorIPCaller

An API commonly used to tamper with resources in a Kubernetes cluster was invoked from a Tor exit node IP address.

Default severity: High

- **Feature:** EKS audit logs

This finding informs you that an API was invoked from a Tor exit node IP address. The API observed is commonly associated with impact tactics where an adversary is trying to manipulate, interrupt, or destroy data within your AWS environment. Tor is software for enabling anonymous communication. It encrypts and randomly bounces communications through relays between a series of network nodes. The last Tor node is called the exit node. This can indicate unauthorized access to your Kubernetes cluster with the intent of hiding the adversary's true identity.

Remediation recommendations:

If the user reported in the finding under the `KubernetesUserDetails` section is `system:anonymous`, investigate why the anonymous user was permitted to invoke the API and revoke the permissions, if needed, by following the instructions in Security best practices for Amazon EKS in the *Amazon EKS User Guide*. If the user is an authenticated user, investigate to determine if the activity was legitimate or malicious. If the activity was malicious revoke access of the user and reverse any changes made by an adversary to your cluster. For more information, see Remediating EKS Protection findings.

Persistence:Kubernetes/ContainerWithSensitiveMount

A container was launched with a sensitive external host path mounted inside.

Default severity: Medium

- **Feature:** EKS audit logs

This finding informs you that a container was launched with a configuration that included a sensitive host path with write access in the `volumeMounts` section. This makes the sensitive host path accessible and writable from inside the container. This technique is commonly used by adversaries to gain access to the host's filesystem.

Remediation recommendations:

If this container launch is unexpected, the credentials of the user identity used to launch the container may be compromised. Revoke access of the user and reverse any changes made by an adversary to your cluster. For more information, see Remediating EKS Protection findings.

If this container launch is expected, it's recommended that you use a suppression rule consisting of a filter criteria based on the `resource.KubernetesDetails.KubernetesWorkloadDetails.containers.imagePrefix` field. In the filter criteria the `imagePrefix` field should be same as the `imagePrefix` specified in the finding. To learn more about creating suppression rules see Suppression rules.

Persistence:Kubernetes/MaliciousIPCaller

An API commonly used to obtain persistent access to a Kubernetes cluster was invoked from a known malicious IP address.

Default severity: Medium

- **Feature:** EKS audit logs

This finding informs you that an API operation was invoked from an IP address that is associated with known malicious activity. The API observed is commonly associated with persistence tactics where an adversary has gained access to your Kubernetes cluster and is attempting to maintain that access.

Remediation recommendations:

If the user reported in the finding under the `KubernetesUserDetails` section is `system:anonymous`, investigate why the anonymous user was permitted to invoke the API and revoke the permissions, if needed, by following the instructions in Security best practices for Amazon EKS in the *Amazon EKS User Guide*. If the user is an authenticated user, investigate to determine if the activity was legitimate or malicious. If the activity was malicious revoke access of the user and reverse any changes made by an adversary to your cluster. For more information, see Remediating EKS Protection findings.

Persistence:Kubernetes/MaliciousIPCaller.Custom

An API commonly used to obtain persistent access to a Kubernetes cluster was invoked from an IP address on a custom threat list.

Default severity: Medium

- **Feature:** EKS audit logs

This finding informs you that an API operation was invoked from an IP address that is included on a threat list that you uploaded. The threat list associated with this finding is listed in the **Additional Information** section of a finding's details. The API observed is commonly associated with persistence tactics where an adversary has gained access to your Kubernetes cluster and is attempting to maintain that access.

Remediation recommendations:

If the user reported in the finding under the `KubernetesUserDetails` section is `system:anonymous`, investigate why the anonymous user was permitted to invoke the API and revoke the permissions, if needed, by following the instructions in Security best practices for Amazon EKS in the *Amazon EKS User Guide*. If the user is an authenticated user, investigate to determine if the activity was legitimate or malicious. If the activity was malicious revoke access of the user and reverse any changes made by an adversary to your cluster. For more information, see Remediating EKS Protection findings.

Persistence:Kubernetes/SuccessfulAnonymousAccess

An API commonly used to obtain high-level permissions to a Kubernetes cluster was invoked by an unauthenticated user.

Default severity: High

- **Feature:** EKS audit logs

This finding informs you that an API operation was successfully invoked by the `system:anonymous` user. API calls made by `system:anonymous` are unauthenticated. The observed API is commonly associated with the persistence tactics where an adversary has gained access to your cluster and is attempting to maintain that access. This activity indicates that anonymous or unauthenticated access is permitted on the API action reported in the finding and may be permitted on other actions. If this behavior is not expected, it may indicate a configuration mistake or that your credentials are compromised.

Remediation recommendations:

You should examine the permissions that have been granted to the `system:anonymous` user on your cluster and ensure that all the permissions are needed. If the permissions were granted mistakenly or maliciously, you should revoke access of the user and reverse any changes made by an adversary to your cluster. For more information, see Security best practices for Amazon EKS in the *Amazon EKS User Guide*.

For more information, see Remediating EKS Protection findings.

Persistence:Kubernetes/TorIPCaller

An API commonly used to obtain persistent access to a Kubernetes cluster was invoked from a Tor exit node IP address.

Default severity: Medium

- **Feature:** EKS audit logs

This finding informs you that an API was invoked from a Tor exit node IP address. The API observed is commonly associated with persistence tactics where an adversary has gained access to your Kubernetes cluster and is attempting to maintain that access. Tor is software for enabling anonymous communication. It encrypts and randomly bounces communications through relays between a series of network nodes. The last Tor node is called the exit node. This can indicate unauthorized access to your AWS resources with the intent of hiding the attacker's true identity.

Remediation recommendations:

If the user reported in the finding under the `KubernetesUserDetails` section is `system:anonymous`, investigate why the anonymous user was permitted to invoke the API and revoke the permissions, if needed, by following the instructions in Security best practices for Amazon EKS in the *Amazon EKS User Guide*. If the user is an authenticated user, investigate to determine if the activity was legitimate or malicious. If the activity was malicious revoke access of the user and reverse any changes made by an adversary to your cluster. For more information, see Remediating EKS Protection findings.

Policy:Kubernetes/AdminAccessToDefaultServiceAccount

The default service account was granted admin privileges on a Kubernetes cluster.

Default severity: High

- **Feature:** EKS audit logs

This finding informs you that the default service account for a namespace in your Kubernetes cluster was granted admin privileges. Kubernetes creates a default service account for all the namespaces in the cluster. It automatically assigns the default service account as an identity to pods that have not been explicitly associated to another service account. If the default service account has admin privileges, it may result in pods being unintentionally launched with admin privileges. If this behavior is not expected, it may indicate a configuration mistake or that your credentials are compromised.

Remediation recommendations:

You should not use the default service account to grant permissions to pods. Instead you should create a dedicated service account for each workload and grant permission to that account on a needs basis. To fix this issue, you should create dedicated service accounts for all your pods and workloads and update the pods and workloads to migrate from the default service account to their dedicated accounts. Then you should remove the admin permission from the default service account. For more information, see [Remediating EKS Protection findings].

Policy:Kubernetes/AnonymousAccessGranted

The `system:anonymous` user was granted API permission on a Kubernetes cluster.

Default severity: High

- **Feature:** EKS audit logs

This finding informs you that a user on your Kubernetes cluster successfully created a `ClusterRoleBinding` or `RoleBinding` to bind the user `system:anonymous` to a role. This enables unauthenticated access to the API operations permitted by the role. If this behavior is not expected, it may indicate a configuration mistake or that your credentials are compromised

Remediation recommendations:

You should examine the permissions that have been granted to the `system:anonymous` user or `system:unauthenticated` group on your cluster and revoke unnecessary anonymous access. For more information, see [Security best practices for Amazon EKS] in the *Amazon EKS User Guide*. If the permissions were granted maliciously, you should revoke access of the user that granted the permissions and reverse any changes made by an adversary to your cluster. For more information, see [Remediating EKS Protection findings].

Policy:Kubernetes/ExposedDashboard

The dashboard for a Kubernetes cluster was exposed to the internet

Default severity: Medium

- **Feature:** EKS audit logs

This finding informs you that Kubernetes dashboard for your cluster was exposed to the internet by a Load Balancer service. An exposed dashboard makes the management interface of your cluster accessible from the internet and allows adversaries to exploit any authentication and access control gaps that may be present.

Remediation recommendations:

You should ensure that strong authentication and authorization is enforced on Kubernetes Dashboard. You should also implement network access control to restrict access to the dashboard from specific IP addresses.

For more information, see Remediating EKS Protection findings.

Policy:Kubernetes/KubeflowDashboardExposed

The Kubeflow dashboard for a Kubernetes cluster was exposed to the Internet

Default severity: Medium

- **Feature:** EKS audit logs

This finding informs you that **Kubeflow** dashboard for your cluster was exposed to the Internet by a Load Balancer service. An exposed **Kubeflow** dashboard makes the management interface of your **Kubeflow** environment accessible from the Internet and allows adversaries to exploit any authentication and access control gaps that may be present.

Remediation recommendations:

You should ensure that strong authentication and authorization is enforced on **Kubeflow** Dashboard. You should also implement network access control to restrict access to the dashboard from specific IP addresses.

For more information, see Remediating EKS Protection findings.

PrivilegeEscalation:Kubernetes/PrivilegedContainer

A privileged container with root level access was launched on your Kubernetes cluster.

Default severity: Medium

- **Feature:** EKS audit logs

This finding informs you that a privileged container was launched on your Kubernetes cluster using an image has never before been used to launch privileged containers in your cluster. A privileged container has root level access to the host. Adversaries can launch privileged containers as a privilege escalation tactic to gain access to and then compromise the host.

Remediation recommendations:

If this container launch is unexpected, the credentials of the user identity used to launch the container may be compromised. Revoke access of the user and reverse any changes made by an adversary to your cluster. For more information, see Remediating EKS Protection findings.

CredentialAccess:Kubernetes/AnomalousBehavior.SecretsAccessed

A Kubernetes API commonly used to access secrets was invoked in an anomalous way.

Default severity: Medium

- **Feature:** EKS audit logs

This finding informs you that an anomalous API operation to retrieve sensitive cluster secrets was invoked by a Kubernetes user in your cluster. The observed API is commonly associated with credential access tactics that can lead to privileged escalation and further access within your cluster. If this behavior is not expected, it may indicate either a configuration mistake or that your AWS credentials are compromised.

The observed API was identified as anomalous by the GuardDuty anomaly detection machine learning (ML) model. The ML model evaluates all user API activity within your EKS cluster and identifies anomalous events that are associated with techniques used by unauthorized users. The ML model tracks multiple factors of the API operation such as the user making the request, the location the request was made from, user agent used, and the namespace that the user operated. You can find the details of the API request that are unusual, in the finding details panel in the GuardDuty console.

Remediation recommendations:

Examine the permissions granted to the Kubernetes user in your cluster and ensure that all these permissions are needed. If the permissions were granted mistakenly or maliciously, revoke user access and reverse any changes made by an unauthorized user to your cluster. For more information, see Remediating EKS Protection findings.

If your AWS credentials are compromised, see Remediating potentially compromised AWS credentials.

PrivilegeEscalation:Kubernetes/AnomalousBehavior.RoleBindingCreated

A RoleBinding or ClusterRoleBinding to an overly permissive role or sensitive namespace was created or modified in your Kubernetes cluster.

Default severity: Medium*

> **Note**
>
> This finding's default severity is Medium. However, if a RoleBinding or ClusterRoleBinding involves the ClusterRoles `admin` or `cluster-admin`, the severity is High.

- **Feature:** EKS audit logs

This finding informs you that a user in your Kubernetes cluster created a `RoleBinding` or `ClusterRoleBinding` to bind a user to a role with admin permissions or sensitive namespaces. If this behavior is not expected, it may indicate either a configuration mistake or that your AWS credentials are compromised.

The observed API was identified as anomalous by the GuardDuty anomaly detection machine learning (ML) model. The ML model evaluates all user API activity within your EKS cluster. This ML model also identifies anomalous events that are associated with the techniques used by an unauthorized user. The ML model also tracks multiple factors of the API operation, such as the user making the request, the location the request was made from, the user agent used, and the namespace that the user operated. You can find the details of the API request that are unusual, in the finding details panel in the GuardDuty console.

Remediation recommendations:

Examine the permissions granted to the Kubernetes user. These permissions are defined in the role and subjects involved in `RoleBinding` and `ClusterRoleBinding`. If the permissions were granted mistakenly or maliciously, revoke user access and reverse any changes made by an unauthorized user to your cluster. For more information, see Remediating EKS Protection findings.

If your AWS credentials are compromised, see Remediating potentially compromised AWS credentials.

Execution:Kubernetes/AnomalousBehavior.ExecInPod

A command was executed inside a pod in an anomalous way.

Default severity: Medium

- **Feature:** EKS audit logs

This finding informs you that a command was executed in a pod using the Kubernetes exec API. The Kubernetes exec API allows running arbitrary commands in a pod. If this behavior is not expected for the user, namespace, or pod, it may indicate either a configuration mistake or that your AWS credentials are compromised.

The observed API was identified as anomalous by the GuardDuty anomaly detection machine learning (ML) model. The ML model evaluates all user API activity within your EKS cluster. This ML model also identifies anomalous events that are associated with the techniques used by an unauthorized user. The ML model also tracks multiple factors of the API operation, such as the user making the request, the location the request was made from, the user agent used, and the namespace that the user operated. You can find the details of the API request that are unusual, in the finding details panel in the GuardDuty console.

Remediation recommendations:

If the execution of this command is unexpected, the credentials of the user identity used to execute the command may have been compromised. Revoke user access and reverse any changes made by an unauthorized user to your cluster. For more information, see Remediating EKS Protection findings.

If your AWS credentials are compromised, see Remediating potentially compromised AWS credentials.

PrivilegeEscalation:Kubernetes/AnomalousBehavior.WorkloadDeployed!PrivilegedContainer

A workload was launched with a privileged container in an anomalous way.

Default severity: High

- **Feature:** EKS audit logs

This finding informs you that a workload was launched with a privileged container in your Amazon EKS cluster. A privileged container has root level access to the host. Unauthorized users can launch privileged containers as a privilege escalation tactic to first gain access to the host and then compromise it.

The observed container creation or modification was identified as anomalous by the GuardDuty anomaly detection machine learning (ML) model. The ML model evaluates all user API and container image activity within your EKS cluster. This ML model also identifies anomalous events that are associated with the techniques used by an unauthorized user. The ML model also tracks multiple factors of the API operation, such as the user making the request, the location the request was made from, the user agent used, container images observed in your account, and the namespace that the user operated. You can find the details of the API request that are unusual, in the finding details panel in the GuardDuty console.

Remediation recommendations:

If this container launch is unexpected, the credentials of the user identity used to launch the container may have been compromised. Revoke user access and reverse any changes made by an unauthorized user to your cluster. For more information, see Remediating EKS Protection findings.

If your AWS credentials are compromised, see Remediating potentially compromised AWS credentials.

If this container launch is expected, it is recommended that you use a suppression rule with a filter criteria based on the `resource.KubernetesDetails.KubernetesWorkloadDetails.containers.imagePrefix` field. In the filter criteria, the `imagePrefix` field must have the same value as the `imagePrefix` field specified in the finding. For more information, see Suppression rules in GuardDuty.

Persistence:Kubernetes/AnomalousBehavior.WorkloadDeployed!ContainerWithSensitiveMount

A workload was deployed in an anomalous way, with a sensitive host path mounted inside the workload.

Default severity: High

- **Feature:** EKS audit logs

This finding informs you that a workload was launched with a container that included a sensitive host path in the `volumeMounts` section. This potentially makes the sensitive host path accessible and writable from inside the container. This technique is commonly used by unauthorized users to gain access to the host's file system.

The observed container creation or modification was identified as anomalous by the GuardDuty anomaly detection machine learning (ML) model. The ML model evaluates all user API and container image activity within your EKS cluster. This ML model also identifies anomalous events that are associated with the techniques used by an unauthorized user. The ML model also tracks multiple factors of the API operation, such as the user making the request, the location the request was made from, the user agent used, container images observed in your account, and the namespace that the user operated. You can find the details of the API request that are unusual, in the finding details panel in the GuardDuty console.

Remediation recommendations:

If this container launch is unexpected, the credentials of the user identity used to launch the container may have been compromised. Revoke user access and reverse any changes made by an unauthorized user to your cluster. For more information, see Remediating EKS Protection findings.

If your AWS credentials are compromised, see Remediating potentially compromised AWS credentials.

If this container launch is expected, it is recommended that you use a suppression rule with a filter criteria based on the `resource.KubernetesDetails.KubernetesWorkloadDetails.containers.imagePrefix` field. In the filter criteria, the `imagePrefix` field must have the same value as the `imagePrefix` field specified in the finding. For more information, see Suppression rules in GuardDuty.

Execution:Kubernetes/AnomalousBehavior.WorkloadDeployed

A workload was launched in an anomalous way.

Default severity: Low*

> **Note**
>
> The default severity is Low. However, if the workload contains a potentially suspicious image name, such as a known pentest tool, or a container running a potentially suspicious command at launch, such as reverse shell commands, then the severity of this finding type will be considered as Medium.

- **Feature:** EKS audit logs

This finding informs you that a Kubernetes workload was created or modified in an anomalous way, such as an API activity, new container images, or risky workload configuration, within your Amazon EKS cluster. Unauthorized users can launch containers as a tactic to execute arbitrary code to first gain access to the host and then compromise it.

The observed container creation or modification was identified as anomalous by the GuardDuty anomaly detection machine learning (ML) model. The ML model evaluates all user API and container image activity within your EKS cluster. This ML model also identifies anomalous events that are associated with the techniques used by an unauthorized user. The ML model also tracks multiple factors of the API operation, such as the user making the request, the location the request was made from, the user agent used, container images observed in your account, and the namespace that the user operated. You can find the details of the API request that are unusual, in the finding details panel in the GuardDuty console.

Remediation recommendations:

If this container launch is unexpected, the credentials of the user identity used to launch the container may have been compromised. Revoke user access and reverse any changes made by an unauthorized user to your cluster. For more information, see Remediating EKS Protection findings.

If your AWS credentials are compromised, see Remediating potentially compromised AWS credentials.

If this container launch is expected, it is recommended that you use a suppression rule with a filter criteria based on the `resource.KubernetesDetails.KubernetesWorkloadDetails.containers.imagePrefix` field. In the filter criteria, the `imagePrefix` field must have the same value as the `imagePrefix` field specified in the finding. For more information, see Suppression rules in GuardDuty.

PrivilegeEscalation:Kubernetes/AnomalousBehavior.RoleCreated

A highly permissive Role or ClusterRole was created or modified in an anomalous way.

Default severity: Low

- **Feature:** EKS audit logs

This finding informs you that an anomalous API operation to create a `Role` or `ClusterRole` with excessive permissions was called by a Kubernetes user in your Amazon EKS cluster. Actors can use role creation with powerful permissions to avoid using built-in admin-like roles and avoid detection. The excessive permissions can lead to privileged escalation, remote code execution, and potentially control over a namespace or cluster. If this behavior is not expected, it may indicate either a configuration mistake or that your credentials are compromised.

The observed API was identified as anomalous by the GuardDuty anomaly detection machine learning (ML) model. The ML model evaluates all user API activity within your Amazon EKS cluster and identifies anomalous events that are associated with the techniques used by unauthorized users. The ML model also tracks multiple factors of the API operation, such as the user making the request, the location the request was made from, the user agent used, container images observed in your account, and the namespace that the user operated. You can find the details of the API request that are unusual, in the finding details panel in the GuardDuty console.

Remediation recommendations:

Examine the permissions defined in `Role` or `ClusterRole` to ensure that all the permissions are needed and follow least privilege principles. If the permissions were granted mistakenly or maliciously, revoke user access and reverse any changes made by an unauthorized user to your cluster. For more information, see Remediating EKS Protection findings.

If your AWS credentials are compromised, see Remediating potentially compromised AWS credentials.

Discovery:Kubernetes/AnomalousBehavior.PermissionChecked

A user checked their access permission in an anomalous way.

Default severity: Low

- **Feature:** EKS audit logs

This finding informs you that a user in your Kubernetes cluster successfully checked whether or not the known powerful permissions that can lead to privileged escalation and remote code execution, are allowed. For example, a common command used to check permissions for a user is `kubectl auth can-i`. If this behavior is not expected, it may indicate either a configuration mistake or that your credentials have been compromised.

The observed API was identified as anomalous by the GuardDuty anomaly detection machine learning (ML) model. The ML model evaluates all user API activity within your Amazon EKS cluster and identifies anomalous events that are associated with the techniques used by unauthorized users. The ML model also tracks multiple factors of the API operation, such as the user making the request, the location the request was made from, permission being checked, and the namespace that the user operated. You can find the details of the API request that are unusual, in the finding details panel in the GuardDuty console.

Remediation recommendations:

Examine the permissions granted to the Kubernetes user to ensure that all the permissions are needed. If the permissions were granted mistakenly or maliciously, revoke user access and reverse any changes made by an unauthorized user to your cluster. For more information, see Remediating EKS Protection findings.

If your AWS credentials are compromised, see Remediating potentially compromised AWS credentials.

GuardDuty Runtime Monitoring finding types

Amazon GuardDuty generates the following Runtime Monitoring findings to indicate potential threats based on the operating system-level behavior from Amazon EC2 hosts and containers in your Amazon EKS clusters, Fargate and Amazon ECS workloads, and Amazon EC2 instances.

> **Note**
>
> Runtime Monitoring finding types are based on the runtime logs collected from hosts. The logs contain fields such as file paths that may be controlled by a malicious actor. These fields are also included in GuardDuty findings to provide runtime context. When processing Runtime Monitoring findings outside of GuardDuty console, you must sanitize finding fields. For example, you can HTML encode finding fields when displaying them on a webpage.

Topics

- CryptoCurrency:Runtime/BitcoinTool.B
- Backdoor:Runtime/C&CActivity.B
- UnauthorizedAccess:Runtime/TorRelay
- UnauthorizedAccess:Runtime/TorClient
- Trojan:Runtime/BlackholeTraffic
- Trojan:Runtime/DropPoint
- CryptoCurrency:Runtime/BitcoinTool.B!DNS
- Backdoor:Runtime/C&CActivity.B!DNS
- Trojan:Runtime/BlackholeTraffic!DNS
- Trojan:Runtime/DropPoint!DNS
- Trojan:Runtime/DGADomainRequest.C!DNS
- Trojan:Runtime/DriveBySourceTraffic!DNS
- Trojan:Runtime/PhishingDomainRequest!DNS
- Impact:Runtime/AbusedDomainRequest.Reputation
- Impact:Runtime/BitcoinDomainRequest.Reputation
- Impact:Runtime/MaliciousDomainRequest.Reputation

- Impact:Runtime/SuspiciousDomainRequest.Reputation
- UnauthorizedAccess:Runtime/MetadataDNSRebind
- Execution:Runtime/NewBinaryExecuted
- PrivilegeEscalation:Runtime/DockerSocketAccessed
- PrivilegeEscalation:Runtime/RuncContainerEscape
- PrivilegeEscalation:Runtime/CGroupsReleaseAgentModified
- DefenseEvasion:Runtime/ProcessInjection.Proc
- DefenseEvasion:Runtime/ProcessInjection.Ptrace
- DefenseEvasion:Runtime/ProcessInjection.VirtualMemoryWrite
- Execution:Runtime/ReverseShell
- DefenseEvasion:Runtime/FilelessExecution
- Impact:Runtime/CryptoMinerExecuted
- Execution:Runtime/NewLibraryLoaded
- PrivilegeEscalation:Runtime/ContainerMountsHostDirectory
- PrivilegeEscalation:Runtime/UserfaultfdUsage
- Execution:Runtime/SuspiciousTool
- Execution:Runtime/SuspiciousCommand
- DefenseEvasion:Runtime/SuspiciousCommand
- DefenseEvasion:Runtime/PtraceAntiDebugging
- Execution:Runtime/MaliciousFileExecuted
- Execution:Runtime/SuspiciousShellCreated
- PrivilegeEscalation:Runtime/ElevationToRoot
- Discovery:Runtime/SuspiciousCommand
- Persistence:Runtime/SuspiciousCommand
- PrivilegeEscalation:Runtime/SuspiciousCommand

CryptoCurrency:Runtime/BitcoinTool.B

An Amazon EC2 instance or a container is querying an IP address that is associated with a cryptocurrency-related activity.

Default severity: High

- **Feature:** Runtime Monitoring

This finding informs you that a process running on the listed EC2 instance or a container in your AWS environment is querying an IP address that is associated with a cryptocurrency-related activity. Threat actors may seek to take control over compute resources to maliciously repurpose them for unauthorized cryptocurrency mining.

The GuardDuty runtime agent monitors events from multiple resource types. To identify the potentially compromised resource, view **Resource type** in the findings panel in the GuardDuty console. Additional context, including process and process lineage information, is available in the finding for further investigation.

Remediation recommendations:

If you use this EC2 instance or a container to mine or manage cryptocurrency, or either of these is otherwise involved in blockchain activity, the CryptoCurrency:Runtime/BitcoinTool.B finding could represent expected activity for your environment. If this is the case in your AWS environment, we recommend that you set up a suppression rule for this finding. The suppression rule should consist of two filter criteria. The first filter criterion should use the **Finding type** attribute with a value of `CryptoCurrency:Runtime/BitcoinTool.B`. The second filter criterion should be the **Instance ID** of the instance or the **Container Image ID** of the container involved in cryptocurrency or blockchain-related activity. For more information, see Suppression rules.

If this activity is unexpected, your resource might have been compromised. For more information, see Remediating Runtime Monitoring findings.

Backdoor:Runtime/C&CActivity.B

An Amazon EC2 instance or a container is querying an IP that is associated with a known command and control server.

Default severity: High

- **Feature:** Runtime Monitoring

This finding informs you that a process running on the listed EC2 instance or a container within your AWS environment is querying an IP address associated with a known command and control

(C&C) server. The listed instance or container might be potentially compromised. Command and control servers are computers that issue commands to members of a botnet.

A botnet is a collection of internet-connected devices that might include PCs, servers, mobile devices, and Internet of Things devices, that are infected and controlled by a common type of malware. Botnets are often used to distribute malware and gather misappropriated information, such as credit card numbers. Depending on the purpose and structure of the botnet, the C&C server might also issue commands to begin a distributed denial of service (DDoS) attack.

> **Note**
>
> If the IP queried is log4j-related, then the fields of the associated finding will include the following values:
>
> - `service.additionalInfo.threatListName = Amazon`
> - `service.additionalInfo.threatName = Log4j Related`

The GuardDuty runtime agent monitors events from multiple resource types. To identify the potentially compromised resource, view **Resource type** in the findings panel in the GuardDuty console. Additional context, including process and process lineage information, is available in the finding for further investigation.

Remediation recommendations:

If this activity is unexpected, your resource might have been compromised. For more information, see [Remediating Runtime Monitoring findings](#).

UnauthorizedAccess:Runtime/TorRelay

Your Amazon EC2 instance or a container is making connections to a Tor network as a Tor relay.

Default severity: High

- **Feature:** Runtime Monitoring

This finding informs you that a process running on the listed EC2 instance or a container in your AWS environment is making connections to a Tor network in a manner that suggests that it's acting as a Tor relay. Tor is software for enabling anonymous communication. Tor increases anonymity of communication by forwarding the client's possibly illicit traffic from one Tor relay to another.

The GuardDuty runtime agent monitors events from multiple resource types. To identify the potentially compromised resource, view **Resource type** in the findings panel in the GuardDuty console.

The GuardDuty runtime agent monitors events from multiple resource types. To identify the potentially compromised resource, view **Resource type** in the findings panel in the GuardDuty console. Additional context, including process and process lineage information, is available in the finding for further investigation.

Remediation recommendations:

If this activity is unexpected, your resource might have been compromised. For more information, see Remediating Runtime Monitoring findings.

UnauthorizedAccess:Runtime/TorClient

Your Amazon EC2 instance or a container is making connections to a Tor Guard or an Authority node.

Default severity: High

- **Feature:** Runtime Monitoring

This finding informs you that a process running on the listed EC2 instance or a container in your AWS environment is making connections to a Tor Guard or an Authority node. Tor is software for enabling anonymous communication. Tor Guards and Authority nodes act as initial gateways into a Tor network. This traffic can indicate that this EC2 instance or the container has been potentially compromised and is acting as a client on a Tor network. This finding may indicate unauthorized access to your AWS resources with the intent of hiding the attacker's true identity.

The GuardDuty runtime agent monitors events from multiple resource types. To identify the potentially compromised resource, view **Resource type** in the findings panel in the GuardDuty console.

The GuardDuty runtime agent monitors events from multiple resource types. To identify the potentially compromised resource, view **Resource type** in the findings panel in the GuardDuty console. Additional context, including process and process lineage information, is available in the finding for further investigation.

Remediation recommendations:

If this activity is unexpected, your resource might have been compromised. For more information, see Remediating Runtime Monitoring findings.

Trojan:Runtime/BlackholeTraffic

An Amazon EC2 instance or a container is attempting to communicate with an IP address of a remote host that is a known black hole.

Default severity: Medium

- **Feature:** Runtime Monitoring

This finding informs you that a process running on the listed EC2 instance or a container in your AWS environment might be compromised because it is trying to communicate with an IP address of a black hole (or sink hole). Black holes are places in the network where incoming or outgoing traffic is silently discarded without informing the source that the data didn't reach its intended recipient. A black hole IP address specifies a host machine that is not running or an address to which no host has been assigned.

The GuardDuty runtime agent monitors events from multiple resource types. To identify the potentially compromised resource, view **Resource type** in the findings panel in the GuardDuty console. Additional context, including process and process lineage information, is available in the finding for further investigation.

Remediation recommendations:

If this activity is unexpected, your resource might have been compromised. For more information, see Remediating Runtime Monitoring findings.

Trojan:Runtime/DropPoint

An Amazon EC2 instance or a container is attempting to communicate with an IP address of a remote host that is known to hold credentials and other stolen data captured by malware.

Default severity: Medium

- **Feature:** Runtime Monitoring

This finding informs you that a process running on the listed EC2 instance or a container in your AWS environment is trying to communicate with an IP address of a remote host that is known to hold credentials and other stolen data captured by malware.

The GuardDuty runtime agent monitors events from multiple resource types. To identify the potentially compromised resource, view **Resource type** in the findings panel in the GuardDuty console. Additional context, including process and process lineage information, is available in the finding for further investigation.

Remediation recommendations:

If this activity is unexpected, your resource might have been compromised. For more information, see Remediating Runtime Monitoring findings.

CryptoCurrency:Runtime/BitcoinTool.B!DNS

An Amazon EC2 instance or a container is querying a domain name that is associated with a cryptocurrency activity.

Default severity: High

- **Feature:** Runtime Monitoring

This finding informs you that a process running on the listed EC2 instance or a container in your AWS environment is querying a domain name that is associated with Bitcoin or other cryptocurrency-related activity. Threat actors may seek to take control over the compute resources in order to maliciously repurpose them for unauthorized cryptocurrency mining.

The GuardDuty runtime agent monitors events from multiple resource types. To identify the potentially compromised resource, view **Resource type** in the findings panel in the GuardDuty console. Additional context, including process and process lineage information, is available in the finding for further investigation.

Remediation recommendations:

If you use this EC2 instance or container to mine or manage cryptocurrency, or either of these is otherwise involved in blockchain activity, the CryptoCurrency:Runtime/BitcoinTool.B!DNS finding could be an expected activity for your environment. If this is the case in your AWS environment, we recommend that you set up a suppression rule for this finding. The suppression rule should consist of two filter criterion. The first criteria should use the **Finding type** attribute with a value of `CryptoCurrency:Runtime/BitcoinTool.B!DNS`. The second filter criteria should be the **Instance ID** of the instance or the **Container Image ID** of the container involved in cryptocurrency or blockchain activity. For more information, see Suppression Rules.

If this activity is unexpected, your resource might have been compromised. For more information, see Remediating Runtime Monitoring findings.

Backdoor:Runtime/C&CActivity.B!DNS

An Amazon EC2 instance or a container is querying a domain name that is associated with a known command and control server.

Default severity: High

- **Feature:** Runtime Monitoring

This finding informs you that a process running on the listed EC2 instance or the container within your AWS environment is querying a domain name associated with a known command and control (C&C) server. The listed EC2 instance or the container might be compromised. Command and control servers are computers that issue commands to members of a botnet.

A botnet is a collection of internet-connected devices which might include PCs, servers, mobile devices, and Internet of Things devices, that are infected and controlled by a common type of malware. Botnets are often used to distribute malware and gather misappropriated information, such as credit card numbers. Depending on the purpose and structure of the botnet, the C&C server might also issue commands to begin a distributed denial of service (DDoS) attack.

Note

If the domain name queried is log4j-related, then the fields of the associated finding will include the following values:

- `service.additionalInfo.threatListName = Amazon`

- service.additionalInfo.threatName = Log4j Related

Note

To test how GuardDuty generates this finding type, you can make a DNS request from your instance (using `dig` for Linux or `nslookup` for Windows) against a test domain `guarddutyc2activityb.com`.

The GuardDuty runtime agent monitors events from multiple resource types. To identify the potentially compromised resource, view **Resource type** in the findings panel in the GuardDuty console. Additional context, including process and process lineage information, is available in the finding for further investigation.

Remediation recommendations:

If this activity is unexpected, your resource might have been compromised. For more information, see Remediating Runtime Monitoring findings.

Trojan:Runtime/BlackholeTraffic!DNS

An Amazon EC2 instance or a container is querying a domain name that is being redirected to a black hole IP address.

Default severity: Medium

- **Feature:** Runtime Monitoring

This finding informs you that a process running on the listed EC2 instance or the container in your AWS environment might be compromised because it is querying a domain name that is being redirected to a black hole IP address. Black holes are places in the network where incoming or outgoing traffic is silently discarded without informing the source that the data didn't reach its intended recipient.

The GuardDuty runtime agent monitors events from multiple resource types. To identify the potentially compromised resource, view **Resource type** in the findings panel in the GuardDuty console. Additional context, including process and process lineage information, is available in the finding for further investigation.

If this activity is unexpected, your resource might have been compromised. For more information, see [Remediating Runtime Monitoring findings](#).

Trojan:Runtime/DropPoint!DNS

An Amazon EC2 instance or a container is querying a domain name of a remote host that is known to hold credentials and other stolen data captured by malware.

Default severity: Medium

- **Feature:** Runtime Monitoring

This finding informs you that a process running on the listed EC2 instance or a container in your AWS environment is querying a domain name of a remote host that is known to hold credentials and other stolen data captured by malware.

The GuardDuty runtime agent monitors events from multiple resource types. To identify the potentially compromised resource, view **Resource type** in the findings panel in the GuardDuty console. Additional context, including process and process lineage information, is available in the finding for further investigation.

Remediation recommendations:

If this activity is unexpected, your resource might have been compromised. For more information, see [Remediating Runtime Monitoring findings](#).

Trojan:Runtime/DGADomainRequest.C!DNS

An Amazon EC2 instance or a container is querying algorithmically generated domains. Such domains are commonly used by malware and could be an indication of a compromised EC2 instance or a container.

Default severity: High

- **Feature:** Runtime Monitoring

This finding informs you that a process running on the listed EC2 instance or the container in your AWS environment is trying to query domain generation algorithm (DGA) domains. Your resource might have been compromised.

DGAs are used to periodically generate a large number of domain names that can be used as rendezvous points with their command and control (C&C) servers. Command and control servers are computers that issue commands to members of a botnet, which is a collection of internet-connected devices that are infected and controlled by a common type of malware. The large number of potential rendezvous points makes it difficult to effectively shut down botnets because infected computers attempt to contact some of these domain names every day to receive updates or commands.

> **Note**
>
> This finding is based on known DGA domains from GuardDuty threat intelligence feeds.

The GuardDuty runtime agent monitors events from multiple resource types. To identify the potentially compromised resource, view **Resource type** in the findings panel in the GuardDuty console. Additional context, including process and process lineage information, is available in the finding for further investigation.

Remediation recommendations:

If this activity is unexpected, your resource might have been compromised. For more information, see [Remediating Runtime Monitoring findings](#).

Trojan:Runtime/DriveBySourceTraffic!DNS

An Amazon EC2 instance or a container is querying a domain name of a remote host that is a known source of Drive-By download attacks.

Default severity: High

- **Feature:** Runtime Monitoring

This finding informs you that a process running on the listed EC2 instance or the container in your AWS environment might be compromised because it is querying a domain name of a remote host that is a known source of drive-by download attacks. These are unintended downloads of computer software from the internet that can initiate an automatic installation of a virus, spyware, or malware.

The GuardDuty runtime agent monitors events from multiple resource types. To identify the potentially compromised resource, view **Resource type** in the findings panel in the GuardDuty console. Additional context, including process and process lineage information, is available in the finding for further investigation.

Remediation recommendations:

If this activity is unexpected, your resource might have been compromised. For more information, see Remediating Runtime Monitoring findings.

Trojan:Runtime/PhishingDomainRequest!DNS

An Amazon EC2 instance or a container is querying domains involved in phishing attacks.

Default severity: High

- **Feature:** Runtime Monitoring

This finding informs you that a process running on the listed EC2 instance or a container in your AWS environment is trying to query a domain involved in phishing attacks. Phishing domains are set up by someone posing as a legitimate institution in order to induce individuals to provide sensitive data, such as personally identifiable information, banking and credit card details, and passwords. Your EC2 instance or the container might be trying to retrieve sensitive data stored on a phishing website, or it may be attempting to set up a phishing website. Your EC2 instance or the container might be compromised.

The GuardDuty runtime agent monitors events from multiple resource types. To identify the potentially compromised resource, view **Resource type** in the findings panel in the GuardDuty console. Additional context, including process and process lineage information, is available in the finding for further investigation.

Remediation recommendations:

If this activity is unexpected, your resource might have been compromised. For more information, see Remediating Runtime Monitoring findings.

Impact:Runtime/AbusedDomainRequest.Reputation

An Amazon EC2 instance or a container is querying a low reputation domain name that is associated with known abused domains.

Default severity: Medium

- **Feature:** Runtime Monitoring

This finding informs you that a process running on the listed EC2 instance or the container within your AWS environment is querying a low reputation domain name associated with known abused domains or IP addresses. Examples of abused domains are top level domain names (TLDs) and second-level domain names (2LDs) providing free subdomain registrations as well as dynamic DNS providers. Threat actors tend to use these services to register domains for free or at low costs. Low reputation domains in this category may also be expired domains resolving to a registrar's parking IP address and therefore may no longer be active. A parking IP is where a registrar directs traffic for domains that have not been linked to any service. The listed Amazon EC2 instance or the container may be compromised as threat actors commonly use these registrar's or services for C&C and malware distribution.

Low reputation domains are based on a reputation score model. This model evaluates and ranks the characteristics of a domain to determine its likelihood of being malicious.

The GuardDuty runtime agent monitors events from multiple resource types. To identify the potentially compromised resource, view **Resource type** in the findings panel in the GuardDuty console. Additional context, including process and process lineage information, is available in the finding for further investigation.

Remediation recommendations:

If this activity is unexpected, your resource might have been compromised. For more information, see Remediating Runtime Monitoring findings.

Impact:Runtime/BitcoinDomainRequest.Reputation

An Amazon EC2 instance or a container is querying a low reputation domain name that is associated with cryptocurrency-related activity.

Default severity: High

- **Feature:** Runtime Monitoring

This finding informs you that a process running on the listed EC2 instance or the container within your AWS environment is querying a low reputation domain name associated with Bitcoin or other cryptocurrency-related activity. Threat actors may seek to take control over compute resources to maliciously repurpose them for unauthorized cryptocurrency mining.

Low reputation domains are based on a reputation score model. This model evaluates and ranks the characteristics of a domain to determine its likelihood of being malicious.

The GuardDuty runtime agent monitors events from multiple resource types. To identify the potentially compromised resource, view **Resource type** in the findings panel in the GuardDuty console. Additional context, including process and process lineage information, is available in the finding for further investigation.

Remediation recommendations:

If you use this EC2 instance or the container to mine or manage cryptocurrency, or if these resources are otherwise involved in blockchain activity, this finding could represent expected activity for your environment. If this is the case in your AWS environment, we recommend that you set up a suppression rule for this finding. The suppression rule should consist of two filter criteria. The first filter criterion should use the **Finding type** attribute with a value of `Impact:Runtime/BitcoinDomainRequest.Reputation`. The second filter criterion should be the **Instance ID** of the instance or the **Container Image ID** of the container is involved in cryptocurrency or blockchain–related activity. For more information, see [Suppression rules].

If this activity is unexpected, your resource might have been compromised. For more information, see [Remediating Runtime Monitoring findings].

Impact:Runtime/MaliciousDomainRequest.Reputation

An Amazon EC2 instance or a container is querying a low reputation domain that is associated with known malicious domains.

Default severity: High

- **Feature:** Runtime Monitoring

This finding informs you that a process running on the listed EC2 instance or the container within your AWS environment is querying a low reputation domain name associated with known malicious domains or IP addresses. For example, domains may be associated with a known sinkhole IP address. Sinkholed domains are domains that were previously controlled by a threat actor, and requests made to them can indicate the instance is compromised. These domains may also be correlated with known malicious campaigns or domain generation algorithms.

Low reputation domains are based on a reputation score model. This model evaluates and ranks the characteristics of a domain to determine its likelihood of being malicious.

The GuardDuty runtime agent monitors events from multiple resource types. To identify the potentially compromised resource, view **Resource type** in the findings panel in the GuardDuty console. Additional context, including process and process lineage information, is available in the finding for further investigation.

Remediation recommendations:

If this activity is unexpected, your resource might have been compromised. For more information, see Remediating Runtime Monitoring findings.

Impact:Runtime/SuspiciousDomainRequest.Reputation

An Amazon EC2 instance or a container is querying a low reputation domain name that is suspicious in nature due to its age, or low popularity.

Default severity: Low

- **Feature:** Runtime Monitoring

This finding informs you that a process running on the listed EC2 instance or the container within your AWS environment is querying a low reputation domain name that is suspected of being malicious. The observed characteristics of this domain were consistent with previously observed malicious domains. However, our reputation model was unable to definitively relate it to a known threat. These domains are typically newly observed or receive a low amount of traffic.

Low reputation domains are based on a reputation score model. This model evaluates and ranks the characteristics of a domain to determine its likelihood of being malicious.

The GuardDuty runtime agent monitors events from multiple resource types. To identify the potentially compromised resource, view **Resource type** in the findings panel in the GuardDuty console. Additional context, including process and process lineage information, is available in the finding for further investigation.

Remediation recommendations:

If this activity is unexpected, your resource might have been compromised. For more information, see [Remediating Runtime Monitoring findings](https://docs.aws.amazon.com/guardduty/latest/ug/guardduty-remediate-runtime-monitoring.html).

UnauthorizedAccess:Runtime/MetadataDNSRebind

An Amazon EC2 instance or a container is performing DNS lookups that resolve to the instance metadata service.

Default severity: High

- **Feature:** Runtime Monitoring

> **Note**
>
> Presently, this finding type is only supported for AMD64 architecture.

This finding informs you that a process running on the listed EC2 instance or a container in your AWS environment is querying a domain that resolves to the EC2 metadata IP address (169.254.169.254). A DNS query of this kind may indicate that the instance is a target of a DNS rebinding technique. This technique can be used to obtain metadata from an EC2 instance, including the IAM credentials associated with the instance.

DNS rebinding involves tricking an application running on the EC2 instance to load return data from a URL, where the domain name in the URL resolves to the EC2 metadata IP address (`169.254.169.254`). This causes the application to access EC2 metadata and possibly make it available to the attacker.

It is possible to access EC2 metadata using DNS rebinding only if the EC2 instance is running a vulnerable application that allows injection of URLs, or if someone accesses the URL in a web browser running on the EC2 instance.

The GuardDuty runtime agent monitors events from multiple resource types. To identify the potentially compromised resource, view **Resource type** in the findings panel in the GuardDuty console. Additional context, including process and process lineage information, is available in the finding for further investigation.

Remediation recommendations:

In response to this finding, you should evaluate if there is a vulnerable application running on the EC2 instance or on the container, or if someone used a browser to access the domain identified in the finding. If the root cause is a vulnerable application, fix the vulnerability. If someone browsed the identified domain, block the domain or prevent users from accessing it. If you determine this finding was related to either case above, Revoke the session associated with the EC2 instance.

Some AWS customers intentionally map the metadata IP address to a domain name on their authoritative DNS servers. If this is the case in your environment, we recommend that you set up a suppression rule for this finding. The suppression rule should consist of two filter criteria. The first filter criterion should use the **Finding type** attribute with a value of `UnauthorizedAccess:Runtime/MetaDataDNSRebind`. The second filter criterion should be **DNS request domain** or the **Container Image ID** of the container. The **DNS request domain** value should match the domain you have mapped to the metadata IP address (`169.254.169.254`). For information about creating suppression rules, see Suppression rules.

If this activity is unexpected, your resource might have been compromised. For more information, see Remediating Runtime Monitoring findings.

Execution:Runtime/NewBinaryExecuted

A newly created or recently modified binary file in a container has been executed.

Default severity: Medium

- **Feature:** Runtime Monitoring

This finding informs you that a newly created or a recently modified binary file, in a container was executed. It is the best practice to keep containers immutable at runtime, and binary files, scripts, or libraries should not be created or modified during the lifetime of the container. This behavior indicates that a malicious actor that has gained access to the container, has downloaded, and executed malware or other software as part of the potential compromise. Although this type of activity could be an indication of a compromise, it is also a common usage pattern. Therefore, GuardDuty uses mechanisms to identify suspicious instances of this activity and generates this finding type only for suspicious instances.

The GuardDuty runtime agent monitors events from multiple resource types. To identify the potentially compromised resource, view **Resource type** in the findings panel in the GuardDuty console. To identify the modifying process and new binary, view the **Modifying process** details and the **Process** details

The details of the modifying process are included in the `service.runtimeDetails.context.modifyingProcess` field of the finding JSON, or under **Modifying Process** in the finding details panel. For this finding type, the modifying process is `/usr/bin/dpkg`, as identified by the `service.runtimeDetails.context.modifyingProcess.executablePath` field of the finding JSON, or as a part of **Modifying Process** in the finding details panel.

The details of the executed new or modified binary are included in the `service.runtimeDetails.process` of the finding JSON, or the **Process** section under **Runtime details**. For this finding type, the new or modified binary is `/usr/bin/python3.8`, as indicated by `service.runtimeDetails.process.executablePath` (**Executable path**) field.

Remediation recommendations:

If this activity is unexpected, your resource might have been compromised. For more information, see [Remediating Runtime Monitoring findings](#).

PrivilegeEscalation:Runtime/DockerSocketAccessed

A process inside a container is communicating with Docker daemon using Docker socket.

Default severity: Medium

- **Feature:** Runtime Monitoring

The Docker socket is a Unix Domain Socket that Docker daemon (`dockerd`) uses to communicate with its clients. A client can perform various actions, such as creating containers by communicating with Docker daemon through the Docker socket. It is suspicious for a container process to access the Docker socket. A container process can escape the container and get a host-level access by communicating with the Docker socket and creating a privileged container.

The GuardDuty runtime agent monitors events from multiple resource types. To identify the potentially compromised resource, view **Resource type** in the findings panel in the GuardDuty console. Additional context, including process and process lineage information, is available in the finding for further investigation.

Remediation recommendations:

If this activity is unexpected, your resource might have been compromised. For more information, see Remediating Runtime Monitoring findings.

PrivilegeEscalation:Runtime/RuncContainerEscape

A container escape attempt through runC was detected.

Default severity: High

- **Feature:** Runtime Monitoring

RunC is the low-level container runtime that high-level container runtimes, such as Docker and Containerd use to spawn and run containers. RunC is always executed with root privileges because it needs to perform the low-level task of creating a container. A threat actor can gain host-level access by either modifying or exploiting a vulnerability in runC binary.

This finding detects modification of runC binary and potential attempts to exploit the following runC vulnerabilities:

- CVE-2019-5736 – Exploitation of CVE-2019-5736 involves overwriting the runC binary from within a container. This finding gets invoked when runC binary is modified by a process inside a container.

- [CVE-2024-21626](https://) – Exploitation of CVE-2024-21626 involves setting the current working directory (CWD) or a container to an open file descriptor `/proc/self/fd/FileDescriptor`. This finding gets invoked when a container process with a current working directory under `/proc/self/fd/` is detected, for example, `/proc/self/fd/7`.

This finding may indicate that a malicious actor has attempted to perform exploitation in one of the following types of containers:

- A new container with an attacker-controlled image.
- An existing container that was accessible to the actor with write permissions on the host level runC binary.

The GuardDuty runtime agent monitors events from multiple resource types. To identify the potentially compromised resource, view **Resource type** in the findings panel in the GuardDuty console.

Remediation recommendations:

If this activity is unexpected, your resource might have been compromised. For more information, see Remediating Runtime Monitoring findings.

PrivilegeEscalation:Runtime/CGroupsReleaseAgentModified

A container escape attempt through CGroups release agent was detected.

Default severity: High

- **Feature:** Runtime Monitoring

This finding informs you that an attempt to modify a control group (cgroup) release agent file has been detected. Linux uses control groups (cgroups) to limit, account for, and isolate the resource usage of a collection of processes. Each cgroup has a release agent file (`release_agent`), a script that Linux executes when any process inside the cgroup terminates. The release agent file is always executed at the host level. A threat actor inside a container can escape to the host by writing arbitrary commands to the release agent file that belongs to a cgroup. When a process inside that cgroup terminates, the commands written by the actor get executed.

The GuardDuty runtime agent monitors events from multiple resource types. To identify the potentially compromised resource, view **Resource type** in the findings panel in the GuardDuty console.

Remediation recommendations:

If this activity is unexpected, your resource might have been compromised. For more information, see Remediating Runtime Monitoring findings.

DefenseEvasion:Runtime/ProcessInjection.Proc

A process injection using proc filesystem was detected in a container or an Amazon EC2 instance.

Default severity: High

- **Feature:** Runtime Monitoring

Process injection is a technique that threat actors use to inject code into processes to evade defenses and potentially elevate privileges. The proc filesystem (procfs) is a special filesystem in Linux that presents the virtual memory of process as a file. The path of that file is `/proc/PID/mem`, where `PID` is the unique ID of the process. A threat actor can write to this file to inject code into the process. This finding identifies potential attempts to write to this file.

The GuardDuty runtime agent monitors events from multiple resource types. To identify the potentially compromised resource, view **Resource type** in the findings panel in the GuardDuty console. Additional context, including process and process lineage information, is available in the finding for further investigation.

Remediation recommendations:

If this activity is unexpected, your resource type might have been compromised. For more information, see Remediating Runtime Monitoring findings.

DefenseEvasion:Runtime/ProcessInjection.Ptrace

A process injection using ptrace system call was detected in a container or an Amazon EC2 instance.

Default severity: Medium

- **Feature:** Runtime Monitoring

Process injection is a technique that threat actors use to inject code into processes to evade defenses and potentially elevate privileges. A process can use ptrace system call to inject code into another process. This finding identifies a potential attempt to inject code into a process using the ptrace system call.

The GuardDuty runtime agent monitors events from multiple resource types. To identify the potentially compromised resource, view **Resource type** in the findings panel in the GuardDuty console. Additional context, including process and process lineage information, is available in the finding for further investigation.

Remediation recommendations:

If this activity is unexpected, your resource type might have been compromised. For more information, see [Remediating Runtime Monitoring findings](#).

DefenseEvasion:Runtime/ProcessInjection.VirtualMemoryWrite

A process injection through a direct write to virtual memory was detected in a container or an Amazon EC2 instance.

Default severity: High

- **Feature:** Runtime Monitoring

Process injection is a technique that threat actors use to inject code into processes to evade defenses and potentially elevate privileges. A process can use a system call such as `process_vm_writev` to directly inject code into another process's virtual memory. This finding identifies a potential attempt to inject code into a process using a system call for writing to the virtual memory of the process.

The GuardDuty runtime agent monitors events from multiple resource types. To identify the potentially compromised resource, view **Resource type** in the findings panel in the GuardDuty console. Additional context, including process and process lineage information, is available in the finding for further investigation.

Remediation recommendations:

If this activity is unexpected, your resource type might have been compromised. For more information, see Remediating Runtime Monitoring findings.

Execution:Runtime/ReverseShell

A process in a container or an Amazon EC2 instance has created a reverse shell.

Default severity: High

- **Feature:** Runtime Monitoring

A reverse shell is a shell session created on a connection that is initiated from the target host to the actor's host. This is opposite to a normal shell that is initiated from the actor's host to the target's host. Threat actors create a reverse shell to execute commands on the target after gaining initial access to the target. This finding identifies potentially suspicious reverse shell connections.

GuardDuty examines related runtime activity and context, and generates this finding type only when the associated activity and context are found to be unusual or suspicious. Additional context, including process and process lineage information, is available in the finding for further investigation.

Remediation recommendations:

The GuardDuty security agent monitors events from multiple sources. To identify the impacted resource, view **Resource type** in the finding details in the GuardDuty console. If this activity is unexpected, your resource type might have been compromised. For more information, see Remediating Runtime Monitoring findings.

DefenseEvasion:Runtime/FilelessExecution

A process in a container or an Amazon EC2 instance is executing code from memory.

Default severity: Medium

- **Feature:** Runtime Monitoring

This finding informs you when a process is executed using an in-memory executable file on disk. This is a common defense evasion technique that avoids writing the malicious executable to the disk to evade file system scanning-based detection. Although this technique is used by malware, it also has some legitimate use cases. One of the examples is a just-in-time (JIT) compiler that writes compiled code to memory and executes it from memory.

The GuardDuty runtime agent monitors events from multiple resource types. To identify the potentially compromised resource, view **Resource type** in the findings panel in the GuardDuty console. Additional context, including process and process lineage information, is available in the finding for further investigation.

Remediation recommendations:

If this activity is unexpected, your resource might have been compromised. For more information, see [Remediating Runtime Monitoring findings].

Impact:Runtime/CryptoMinerExecuted

A container or an Amazon EC2 instance is executing a binary file that is associated with a cryptocurrency mining activity.

Default severity: High

- **Feature:** Runtime Monitoring

This finding informs you that a process running on the listed EC2 instance or container in your AWS environment is executing a binary file that is associated with a cryptocurrency mining activity. Threat actors may seek to take control over compute resources to maliciously repurpose them for unauthorized cryptocurrency mining.

The GuardDuty runtime agent monitors events from multiple resource types. To identify the potentially compromised resource, view **Resource type** in the findings panel in the GuardDuty console. Additional context, including process and process lineage information, is available in the finding for further investigation.

Remediation recommendations:

The GuardDuty runtime agent monitors events from multiple resources. To identify the affected resource, view **Resource type** in the findings details in the GuardDuty console and see Remediating Runtime Monitoring findings.

Execution:Runtime/NewLibraryLoaded

A newly created or recently modified library was loaded by a process inside a container.

Default severity: Medium

- **Feature:** Runtime Monitoring

This finding informs you that a library was created or modified inside a container during runtime and loaded by a process running inside the container. The best practice is to keep the containers immutable at the runtime, and not to create or modify the binary files, scripts, or libraries during the lifetime of the container. Loading of a newly created or modified library in a container may indicate suspicious activity. This behavior indicates that a malicious actor has potentially gained access to the container, has downloaded, and executed malware or other software as a part of the potential compromise. Although this type of activity could be an indication of a compromise, it is also a common usage pattern. Therefore, GuardDuty uses mechanisms to identify suspicious instances of this activity and generates this finding type only for suspicious instances.

The GuardDuty runtime agent monitors events from multiple resources. To identify the affected resource, view **Resource type** in the findings details in the GuardDuty console. Additional context, including process and process lineage information, is available in the finding for further investigation.

Remediation recommendations:

If this activity is unexpected, your resource might have been compromised. For more information, see Remediating Runtime Monitoring findings.

PrivilegeEscalation:Runtime/ContainerMountsHostDirectory

A process inside a container mounted a host filesystem at runtime.

Default severity: Medium

- **Feature:** Runtime Monitoring

Multiple container escape techniques involve mounting a host filesystem inside a container at runtime. This finding informs you that a process inside a container potentially attempted to mount a host filesystem, which may indicate an attempt to escape to the host.

The GuardDuty runtime agent monitors events from multiple resources. To identify the affected resource, view **Resource type** in the findings details in the GuardDuty console. Additional context, including process and process lineage information, is available in the finding for further investigation.

Remediation recommendations:

If this activity is unexpected, your resource might have been compromised. For more information, see Remediating Runtime Monitoring findings.

PrivilegeEscalation:Runtime/UserfaultfdUsage

A process used `userfaultfd` system calls to handle page faults in user space.

Default severity: Medium

- **Feature:** Runtime Monitoring

Typically, page faults are handled by the kernel in kernel space. However, `userfaultfd` system call allows a process to handle page faults on a filesystem in user space. This is a useful feature that enables implementation of user-space filesystems. On the other hand, it can also be used by a potentially malicious process to interrupt kernel from user space. Interrupting kernel by using `userfaultfd` system call is a common exploitation technique to extend race windows during exploitation of kernel race conditions. Use of `userfaultfd` may indicate suspicious activity on the Amazon Elastic Compute Cloud (Amazon EC2) instance.

The GuardDuty runtime agent monitors events from multiple resources. To identify the affected resource, view **Resource type** in the findings details in the GuardDuty console. Additional context, including process and process lineage information, is available in the finding for further investigation.

Remediation recommendations:

If this activity is unexpected, your resource might have been compromised. For more information, see Remediating Runtime Monitoring findings.

Execution:Runtime/SuspiciousTool

A container or an Amazon EC2 instance is running a binary file or script that is frequently used in offensive security scenarios such as pentesting engagement.

Default severity: Variable

The severity of this finding can be either high or low, depending on whether the detected suspicious tool is considered to be dual-use or is it exclusively for offensive use.

- **Feature:** Runtime Monitoring

This finding informs you that a suspicious tool has been executed on an EC2 instance or container within your AWS environment. This includes tools used in pentesting engagements, also known as backdoor tools, network scanners, and network sniffers. All these tools can be used in benign contexts but are also frequently used by threat actors with malicious intent. Observing offensive security tools could indicate that the associated EC2 instance or container has been compromised.

GuardDuty examines related runtime activity and context so that it generates this finding only when the associated activity and context are potentially suspicious.

The GuardDuty runtime agent monitors events from multiple resources. To identify the affected resource, view **Resource type** in the findings details in the GuardDuty console. When applicable, additional context, including process and process lineage information, is available in the finding for further investigation.

Remediation recommendations:

If this activity is unexpected, your resource might have been compromised. For more information, see Remediating Runtime Monitoring findings.

Execution:Runtime/SuspiciousCommand

A suspicious command has been executed on an Amazon EC2 instance or a container that is indicative of a compromise.

Default severity: Variable

Depending on the impact of the observed malicious pattern, the severity of this finding type could be either low, medium, or high.

- **Feature:** Runtime Monitoring

This finding informs you that a suspicious command has been executed and it indicates that an Amazon EC2 instance or a container in your AWS environment has been compromised. This might mean that either a file was downloaded from a suspicious source and then executed, or a running process displays a known malicious pattern in its command line. This further indicates that malware is running on the system.

GuardDuty examines related runtime activity and context so that it generates this finding only when the associated activity and context are potentially suspicious.

The GuardDuty runtime agent monitors events from multiple resources. To identify the affected resource, view **Resource type** in the findings details in the GuardDuty console. When applicable, additional context, including process and process lineage information, is available in the finding for further investigation.

Remediation recommendations:

If this activity is unexpected, your resource might have been compromised. For more information, see Remediating Runtime Monitoring findings.

DefenseEvasion:Runtime/SuspiciousCommand

A command has been executed on the listed Amazon EC2 instance or a container, it attempts to modify or disable a Linux defense mechanism, such as firewall or essential system services.

Default severity: Variable

Depending on which defense mechanism has been modified or disabled, the severity of this finding type can be either high, medium, or low.

- **Feature:** Runtime Monitoring

This finding informs you that a command that attempts to hide an attack from the local system's security services, has been executed. This includes actions such as disabling the Unix firewall, modifying local IP tables, removing crontab entries, disabling a local service, or taking over the `LDPreload` function. Any modification is highly suspicious and a potential indicator of compromise. Therefore, these mechanisms detect or prevent further compromise of the system.

GuardDuty examines related runtime activity and context so that it generates this finding only when the associated activity and context are potentially suspicious.

The GuardDuty runtime agent monitors events from multiple resources. To identify the potentially compromised resource, view **Resource type** in the findings details in the GuardDuty console. When applicable, additional context, including process and process lineage information, is available in the finding for further investigation.

Remediation recommendations:

If this activity is unexpected, your resource might have been compromised. For more information, see Remediating Runtime Monitoring findings.

DefenseEvasion:Runtime/PtraceAntiDebugging

A process in a container or an Amazon EC2 instance has executed an anti-debugging measure using the ptrace system call.

Default severity: Low

- **Feature:** Runtime Monitoring

This finding shows that a process running on the listed Amazon EC2 instance or a container within your AWS environment has used the ptrace system call with the `PTRACE_TRACEME` option. This activity would cause an attached debugger to detach from the running process. If no debugger is attached, it has no effect. However, the activity in itself raises suspicion. This might indicate that malware is running on the system. Malware frequently uses anti-debugging techniques to evade analysis, and these techniques can be detected at runtime.

GuardDuty examines related runtime activity and context so that it generates this finding only when the associated activity and context are potentially suspicious.

The GuardDuty runtime agent monitors events from multiple resources. To identify the affected resource, view **Resource type** in the findings details in the GuardDuty console. Additional context, including process and process lineage information, is available in the finding for further investigation.

Remediation recommendations:

If this activity is unexpected, your resource might have been compromised. For more information, see Remediating Runtime Monitoring findings.

Execution:Runtime/MaliciousFileExecuted

A known malicious executable file has been executed on an Amazon EC2 instance or a container.

Default severity: High

- **Feature:** Runtime Monitoring

This finding informs you that a known malicious executable has been executed on Amazon EC2 instance or a container within your AWS environment. This is a strong indicator that the instance or container has been potentially compromised and that malware has been executed.

GuardDuty examines related runtime activity and context so that it generates this finding only when the associated activity and context are potentially suspicious.

The GuardDuty runtime agent monitors events from multiple resources. To identify the affected resource, view **Resource type** in the findings details in the GuardDuty console. When applicable, additional context, including process and process lineage information, is available in the finding for further investigation.

Remediation recommendations:

If this activity is unexpected, your resource might have been compromised. For more information, see Remediating Runtime Monitoring findings.

Execution:Runtime/SuspiciousShellCreated

A network service or network-accessible process on an Amazon EC2 instance, or in a container has started an interactive shell process.

Default severity: Low

- **Feature:** Runtime Monitoring

This finding informs you that a network-accessible service on an Amazon EC2 instance or in a container within your AWS environment has launched an interactive shell. Under certain circumstances, this scenario may indicate post-exploitation behavior. Interactive shells allow attackers to execute arbitrary commands on a compromised instance or container.

The GuardDuty runtime agent monitors events from multiple resources. To identify the affected resource, view **Resource type** in the findings details in the GuardDuty console. Additional context, including process and process lineage information, is available in the finding for further investigation. You can view the network-accessible process information in the parent process details.

Remediation recommendations:

If this activity is unexpected, your resource might have been compromised. For more information, see Remediating Runtime Monitoring findings.

PrivilegeEscalation:Runtime/ElevationToRoot

A process running on the listed Amazon EC2 instance or container has assumed root privileges.

Default severity: Medium

- **Feature:** Runtime Monitoring

This finding informs you that a process running on the listed Amazon EC2 or in the listed container within your AWS environment has assumed root privileges through unusual or suspicious `setuid` binary execution. This indicates that a running process has been potentially compromised, for the EC2 instance through an exploit, or through `setuid` exploitation. By using the root privileges, the attacker can potentially execute commands on the instance or the container.

While GuardDuty is designed to not generate this finding type for activities involving regular use of the `sudo` command, it will generate this finding when it identifies the activity as unusual or suspicious.

GuardDuty examines related runtime activity and context, and generates this finding type only when the associated activity and context are unusual or suspicious.

The GuardDuty runtime agent monitors events from multiple resources. To identify the affected resource, view **Resource type** in the findings details in the GuardDuty console. Additional context, including process and process lineage information, is available in the finding for further investigation.

Remediation recommendations:

If this activity is unexpected, your resource might have been compromised. For more information, see Remediating Runtime Monitoring findings.

Discovery:Runtime/SuspiciousCommand

A suspicious command has been executed on an Amazon EC2 instance or in a container, which allows an attacker to gain information about the local system, surrounding AWS infrastructure, or container infrastructure.

Default severity: Low

Feature: Runtime Monitoring

This finding informs you that a process running on the listed Amazon EC2 instance or container in your AWS environment has executed a command that might provide an attacker with crucial information to potentially advance the attack. The following information may have been retrieved:

- Local system such as user or network configuration,
- Other available AWS resources and permissions, or
- Kubernetes infrastructure such as services and pods.

The Amazon EC2 instance or the container that is listed in the finding detail might have been compromised.

The GuardDuty runtime agent monitors events from multiple resource types. To identify the potentially compromised resource, view **Resource type** in the findings details in the GuardDuty console. You can find the details about the suspicious command in the `service.runtimeDetails.context` field of the finding JSON. Additional context, including process and process lineage information, is available in the finding for further investigation.

Remediation recommendations:

If this activity is unexpected, your resource might have been compromised. For more information, see Remediating Runtime Monitoring findings.

Persistence:Runtime/SuspiciousCommand

A suspicious command has been executed on an Amazon EC2 instance or in a container, which allows an attacker to persist access and control in your AWS environment.

Default severity: Medium

- **Feature:** Runtime Monitoring

This finding informs you that a process running on the listed Amazon EC2 instance or in a container within your AWS environment has executed a suspicious command. The command installs a persistence method which allows malware to run uninterruptedly, or allows an attacker to continuously access the potentially compromised instance or container resource type. This could potentially mean that a system service has been installed or modified, the `crontab` has been modified, or a new user has been added to the system configuration.

GuardDuty examines related runtime activity and context, and generates this finding type only when the associated activity and context are unusual or suspicious.

The Amazon EC2 instance or the container that is listed in the finding detail might have been compromised.

The GuardDuty runtime agent monitors events from multiple resources. To identify the potentially compromised resource, view **Resource type** in the findings details in the GuardDuty console. You can find the details about the suspicious command in the `service.runtimeDetails.context` field of the finding JSON. When applicable, additional context, including process and process lineage information, is available in the finding for further investigation.

Remediation recommendations:

If this activity is unexpected, your resource might have been compromised. For more information, see Remediating Runtime Monitoring findings.

PrivilegeEscalation:Runtime/SuspiciousCommand

A suspicious command has been executed on an Amazon EC2 instance or in a container, which allows an attacker to escalate privileges.

Default severity: Medium

- **Feature:** Runtime Monitoring

This finding informs you that a process running on the listed Amazon EC2 instance or in a container within your AWS environment has executed a suspicious command. The command attempts to perform privilege escalation, which allows an adversary to perform high privilege tasks.

GuardDuty examines related runtime activity and context, and generates this finding type only when the associated activity and context are unusual or suspicious.

The Amazon EC2 instance or the container that is listed in the finding detail might have been compromised.

The GuardDuty runtime agent monitors events from multiple resources. To identify the affected resource, view **Resource type** in the findings details in the GuardDuty console. When applicable, additional context, including process and process lineage information, is available in the finding for further investigation.

Remediation recommendations:

If this activity is unexpected, your resource might have been compromised. For more information, see Remediating Runtime Monitoring findings.

Malware Protection for EC2 finding types

GuardDuty Malware Protection for EC2 provides a single Malware Protection for EC2 finding for all threats detected during the scan of an EC2 instance or a container workload. The finding includes the total number of detections made during the scan, and based on the severity, provides details for the top 32 threats that it detects. Unlike other GuardDuty findings, Malware Protection for EC2 findings are not updated when the same EC2 instance or container workload is scanned again.

A new Malware Protection for EC2 finding is generated for each scan that detects malware. Malware Protection for EC2 findings include information about the corresponding scan that produced the finding as well as the GuardDuty finding that initiated this scan. This makes it easier to correlate the suspicious behavior with the detected malware.

> **Note**
>
> When GuardDuty detects malicious activity on a container workload, Malware Protection for EC2 doesn't generate an EC2 level finding.

The following findings are specific to GuardDuty Malware Protection for EC2.

Topics

- Execution:EC2/MaliciousFile
- Execution:ECS/MaliciousFile
- Execution:Kubernetes/MaliciousFile
- Execution:Container/MaliciousFile
- Execution:EC2/SuspiciousFile
- Execution:ECS/SuspiciousFile
- Execution:Kubernetes/SuspiciousFile
- Execution:Container/SuspiciousFile

Execution:EC2/MaliciousFile

A malicious file has been detected on an EC2 instance.

Default severity: Varies depending on the detected threat.

- **Feature:** EBS Malware Protection

This finding indicates that the GuardDuty Malware Protection for EC2 scan has detected one or more malicious files on the listed EC2 instance within your AWS environment. This listed instance might be compromised. For more information, see **Threats detected** section in the findings' details.

Remediation recommendations:

If this activity is unexpected, your instance may be compromised. For more information, see Remediating a potentially compromised Amazon EC2 instance.

Execution:ECS/MaliciousFile

A malicious file has been detected on an ECS cluster.

Default severity: Varies depending on the detected threat.

- **Feature:** EBS Malware Protection

This finding indicates that the GuardDuty Malware Protection for EC2 scan has detected one or more malicious files on a container workload that belongs to an ECS cluster. For more information, see **Threats detected** section in the findings' details.

Remediation recommendations:

If this activity is unexpected, your container belonging to the ECS cluster may be compromised. For more information, see Remediating a potentially compromised ECS cluster.

Execution:Kubernetes/MaliciousFile

A malicious file has been detected on an Kubernetes cluster.

Default severity: Varies depending on the detected threat.

- **Feature:** EBS Malware Protection

This finding indicates that the GuardDuty Malware Protection for EC2 scan has detected one or more malicious files on a container workload that belongs to a Kubernetes cluster. If this is an EKS managed cluster, the findings details will provide additional information about the impacted EKS resource. For more information, see **Threats detected** section in the findings' details.

Remediation recommendations:

If this activity is unexpected, your container workload may be compromised. For more information, see Remediating EKS Protection findings.

Execution:Container/MaliciousFile

A malicious file has been detected on a standalone container.

Default severity: Varies depending on the detected threat.

- **Feature:** EBS Malware Protection

This finding indicates that the GuardDuty Malware Protection for EC2 scan has detected one or more malicious files on a container workload and no cluster information has been identified. For more information, see **Threats detected** section in the findings' details.

Remediation recommendations:

If this activity is unexpected, your container workload may be compromised. For more information, see Remediating a potentially compromised standalone container.

Execution:EC2/SuspiciousFile

A suspicious file has been detected on an EC2 instance.

Default severity: Varies depending on the detected threat.

- **Feature:** EBS Malware Protection

This finding indicates that the GuardDuty Malware Protection for EC2 scan has detected one or more suspicious files on an EC2 instance. For more information, see **Threats detected** section in the findings' details.

`SuspiciousFile` type detections indicate that potentially unwanted programs such as adware, spyware, or dual use tools are present on an impacted resource. These programs could have a negative impact on your resource, or be used by attackers for malicious purposes. For example, networking tools can be used legitimately or maliciously by adversaries as hack tools to try and compromise resources.

When a suspicious file has been detected, evaluate whether you expect to see the detected file in your AWS environment. If the file is unexpected, follow the remediation recommendations provided in the next section.

Remediation recommendations:

If this activity is unexpected, your instance may be compromised. For more information, see Remediating a potentially compromised Amazon EC2 instance.

Execution:ECS/SuspiciousFile

A suspicious file has been detected on an ECS cluster.

Default severity: Varies depending on the detected threat.

- **Feature:** EBS Malware Protection

This finding indicates that the GuardDuty Malware Protection for EC2 scan has detected one or more suspicious files on a container that belongs to an ECS cluster. For more information, see **Threats detected** section in the findings' details.

`SuspiciousFile` type detections indicate that potentially unwanted programs such as adware, spyware, or dual use tools are present on an impacted resource. These programs could have a negative impact on your resource, or be used by attackers for malicious purposes. For example, networking tools can be used legitimately or maliciously by adversaries as hack tools to try and compromise resources.

When a suspicious file has been detected, evaluate whether you expect to see the detected file in your AWS environment. If the file is unexpected, follow the remediation recommendations provided in the next section.

Remediation recommendations:

If this activity is unexpected, your container belonging to the ECS cluster may be compromised. For more information, see Remediating a potentially compromised ECS cluster.

Execution:Kubernetes/SuspiciousFile

A suspicious file has been detected on a Kubernetes cluster.

Default severity: Varies depending on the detected threat.

- **Feature:** EBS Malware Protection

This finding indicates that the GuardDuty Malware Protection for EC2 scan has detected one or more suspicious files on a container that belongs to a Kubernetes cluster. If this is an EKS managed cluster, the findings' details will provide additional information about the impacted EKS. For more information, see **Threats detected** section in the findings' details.

`SuspiciousFile` type detections indicate that potentially unwanted programs such as adware, spyware, or dual use tools are present on an impacted resource. These programs could have a negative impact on your resource, or be used by attackers for malicious purposes. For example, networking tools can be used legitimately or maliciously by adversaries as hack tools to try and compromise resources.

When a suspicious file has been detected, evaluate whether you expect to see the detected file in your AWS environment. If the file is unexpected, follow the remediation recommendations provided in the next section.

Remediation recommendations:

If this activity is unexpected, your container workload may be compromised. For more information, see [Remediating EKS Protection findings](#).

Execution:Container/SuspiciousFile

A suspicious file has been detected on a standalone container.

Default severity: Varies depending on the detected threat.

- **Feature:** EBS Malware Protection

This finding indicates that the GuardDuty Malware Protection for EC2 scan has detected one or more suspicious files on a container with no cluster information. For more information, see **Threats detected** section in the findings' details.

`SuspiciousFile` type detections indicate that potentially unwanted programs such as adware, spyware, or dual use tools are present on an impacted resource. These programs could have a negative impact on your resource, or be used by attackers for malicious purposes. For example, networking tools can be used legitimately or maliciously by adversaries as hack tools to try and compromise resources.

When a suspicious file has been detected, evaluate whether you expect to see the detected file in your AWS environment. If the file is unexpected, follow the remediation recommendations provided in the next section.

Remediation recommendations:

If this activity is unexpected, your container workload may be compromised. For more information, see Remediating a potentially compromised standalone container.

Malware Protection for S3 finding type

GuardDuty generates a finding only when it detects a potential security threat in your AWS account. An Malware Protection for S3 finding indicates that the uploaded object that initiated the malware scan contains a potentially malicious file.

For Amazon GuardDuty to generate a finding in your AWS account, enable both GuardDuty and Malware Protection for S3. The best practice is to first enable GuardDuty and then Malware Protection for S3. If this order is different for you, make sure to enable GuardDuty before an S3 object gets upload to your protected bucket.

> **Note**
> GuardDuty can't generate a finding for an S3 object that was scanned before you enabled GuardDuty. To scan an existing S3 object, you may upload it again.

Object:S3/MaliciousFile

A malicious file has been detected on a scanned S3 object.

Default severity: High

- **Feature:** Malware Protection for S3

This finding indicates that a malware scan has detected the listed S3 object to be malicious. For more information, view the **Threats detected** section in the finding details panel.

Recommendation remediation:

If this finding was unexpected, the S3 object is potentially malicious. For information about recommended remediation steps, see Remediating a potentially malicious S3 object.

GuardDuty RDS Protection finding types

GuardDuty RDS Protection detects anomalous login behavior on your database instance. The following findings are specific to the Supported Amazon Aurora, Amazon RDS, and Aurora Limitless databases and will have a **Resource Type** of `RDSDBInstance` or `RDSLimitlessDB`. The severity and details of the findings will differ based on the finding type.

Topics

- CredentialAccess:RDS/AnomalousBehavior.SuccessfulLogin
- CredentialAccess:RDS/AnomalousBehavior.FailedLogin
- CredentialAccess:RDS/AnomalousBehavior.SuccessfulBruteForce
- CredentialAccess:RDS/MaliciousIPCaller.SuccessfulLogin
- CredentialAccess:RDS/MaliciousIPCaller.FailedLogin
- Discovery:RDS/MaliciousIPCaller
- CredentialAccess:RDS/TorIPCaller.SuccessfulLogin
- CredentialAccess:RDS/TorIPCaller.FailedLogin
- Discovery:RDS/TorIPCaller

CredentialAccess:RDS/AnomalousBehavior.SuccessfulLogin

A user successfully logged into an RDS database in your account in an anomalous way.

Default severity: Variable

> **Note**
>
> Depending on the anomalous behavior associated with this finding, the default severity can Low, Medium, and High.
>
> - **Low –** If the user name associated with this finding logged in from an IP address that is associated with a private network.

- **Medium –** If the user name associated with this finding logged in from a public IP address.
- **High –** If there is a consistent pattern of failed login attempts from public IP addresses indicative of overly permissive access policies.

- **Feature:** RDS login activity monitoring

This finding informs you that an anomalous successful login was observed on an RDS database in your AWS environment. This may indicate that a previous unseen user logged into an RDS database for the first time. A common scenario is an internal user logging into a database that is accessed programmatically by applications and not by individual users.

This successful login was identified as anomalous by the GuardDuty anomaly detection machine learning (ML) model. The ML model evaluates all database login events in your Supported Amazon Aurora, Amazon RDS, and Aurora Limitless databases and identifies anomalous events that are associated with techniques used by adversaries. The ML model tracks various factors of the RDS login activity such as the user that made the request, the location the request was made from, and the specific database connection details that were used. For information about the login events that are potentially unusual, see RDS login activity-based anomalies.

Remediation recommendations:

If this activity is unexpected for the associated database, it is recommended to change the password of the associated database user, and review available audit logs for activity performed by the anomalous user. Medium and high severity findings may indicate that there is an overly permissive access policy to the database, and user credentials may have been exposed or compromised. It is recommended to place the database in a private VPC, and limit the security group rules to allow traffic only from the necessary sources. For more information, see Remediating potentially compromised database with successful login events.

CredentialAccess:RDS/AnomalousBehavior.FailedLogin

One or more unusual failed login attempts were observed on an RDS database in your account.

Default severity: Low

- **Feature:** RDS login activity monitoring

This finding informs you that one or more anomalous failed logins were observed on an RDS database in your AWS environment. A failed login attempts from public IP addresses may indicate that the RDS database in your account has been subject to an attempted brute force attack by a potentially malicious actor.

These failed logins were identified as anomalous by the GuardDuty anomaly detection machine learning (ML) model. The ML model evaluates all database login events in your Supported Amazon Aurora, Amazon RDS, and Aurora Limitless databases and identifies anomalous events that are associated with techniques used by adversaries. The ML model tracks various factors of the RDS login activity such as the user that made the request, the location the request was made from, and the specific database connection details that were used. For information about the RDS login activity that are potentially unusual, see RDS login activity-based anomalies.

Remediation recommendations:

If this activity is unexpected for the associated database, it may indicate that the database is publicly exposed or there is an overly permissive access policy to the database. It is recommended to place the database in a private VPC, and limit the security group rules to allow traffic only from the necessary sources. For more information, see Remediating potentially compromised database with failed login events.

CredentialAccess:RDS/AnomalousBehavior.SuccessfulBruteForce

A user successfully logged into an RDS database in your account from a public IP address in an anomalous way after a consistent pattern of unusual failed login attempts.

Default severity: High

- **Feature:** RDS login activity monitoring

This finding informs you that an anomalous login indicative of a successful brute force was observed on an RDS database in your AWS environment. Prior to an anomalous successful login, a consistent pattern of unusual failed login attempts was observed. This indicates that the user and password associated with the RDS database in your account may have been compromised, and the RDS database may have been accessed by a potentially malicious actor.

This successful brute force login was identified as anomalous by the GuardDuty anomaly detection machine learning (ML) model. The ML model evaluates all database login events in your Supported Amazon Aurora, Amazon RDS, and Aurora Limitless databases and identifies anomalous events that are associated with techniques used by adversaries. The ML model tracks various factors of the RDS login activity such as the user that made the request, the location the request was made from, and the specific database connection details that were used. For information about the RDS login activity that are potentially unusual, see RDS login activity-based anomalies.

Remediation recommendations:

This activity indicates that database credentials may have been exposed or compromised. It is recommended to change the password of the associated database user, and review available audit logs for activity performed by the potentially compromised user. A consistent pattern of unusual failed login attempts indicate an overly permissive access policy to the database or the database may have also been public exposed. It is recommended to place the database in a private VPC, and limit the security group rules to allow traffic only from the necessary sources. For more information, see Remediating potentially compromised database with successful login events.

CredentialAccess:RDS/MaliciousIPCaller.SuccessfulLogin

A user successfully logged into an RDS database in your account from a known malicious IP address.

Default severity: High

- **Feature:** RDS login activity monitoring

This finding informs you that a successful RDS login activity occurred from an IP address that is associated with a known malicious activity in your AWS environment. This indicates that the user and password associated with the RDS database in your account may have been compromised, and the RDS database may have been accessed by a potentially malicious actor.

Remediation recommendations:

If this activity is unexpected for the associated database, it may indicate that the user credentials may have been exposed or compromised. It is recommended to change the password of the associated database user, and review the available audit logs for activity performed by the compromised user. This activity may also indicate that there is an overly permissive access policy to the database or the database is publicly exposed. It is recommended to place the database in a private VPC, and limit the security group rules to allow traffic only from the necessary sources. For more information, see [Remediating potentially compromised database with successful login events].

CredentialAccess:RDS/MaliciousIPCaller.FailedLogin

An IP address that is associated with a known malicious activity unsuccessfully attempted to log in to an RDS database in your account.

Default severity: Medium

- **Feature:** RDS login activity monitoring

This finding informs you that an IP address associated with known malicious activity attempted to log in to an RDS database in your AWS environment, but failed to provide the correct user name or password. This indicates that a potentially malicious actor may be attempting to compromise the RDS database in your account.

Remediation recommendations:

If this activity is unexpected for the associated database, it may indicate that there is an overly permissive access policy to the database or the database is publicly exposed. It is recommended to place the database in a private VPC, and limit the security group rules to allow traffic only from the necessary sources. For more information, see [Remediating potentially compromised database with failed login events].

Discovery:RDS/MaliciousIPCaller

An IP address that is associated with a known malicious activity probed an RDS database in your account; no authentication attempt was made.

Default severity: Medium

- **Feature:** RDS login activity monitoring

This finding informs you that an IP address associated with known a malicious activity probed an RDS database in your AWS environment, though no login attempt was made. This may indicate that a potentially malicious actor is attempting to scan for a publicly accessible infrastructure.

Remediation recommendations:

If this activity is unexpected for the associated database, it may indicate that there is an overly permissive access policy to the database or the database is publicly exposed. It is recommended to place the database in a private VPC, and limit the security group rules to allow traffic only from the necessary sources. For more information, see Remediating potentially compromised database with failed login events.

CredentialAccess:RDS/TorIPCaller.SuccessfulLogin

A user successfully logged into an RDS database in your account from a Tor exit node IP address.

Default severity: High

- **Feature:** RDS login activity monitoring

This finding informs you that a user successfully logged in to an RDS database in your AWS environment, from a Tor exit node IP address. Tor is a software for enabling anonymous communication. It encrypts and randomly bounces communications through relays between a series of network nodes. The last Tor node is called the exit node. This can indicate unauthorized access to the RDS resources in your account, with the intent of hiding the anonymous user's true identity.

Remediation recommendations:

If this activity is unexpected for the associated database, it may indicate that the user credentials may have been exposed or compromised. It is recommended to change the password of the associated database user, and review the available audit logs for activity performed by the compromised user. This activity may also indicate that there is an overly permissive access policy to the database or the database is publicly exposed. It is recommended to place the database in a private VPC, and limit the security group rules to allow traffic only from the necessary sources. For more information, see Remediating potentially compromised database with successful login events.

CredentialAccess:RDS/TorIPCaller.FailedLogin

A Tor IP address attempted to unsuccessfully log in to an RDS database in your account.

Default severity: Medium

- **Feature:** RDS login activity monitoring

This finding informs you that a Tor exit node IP address attempted to log in to an RDS database in your AWS environment, but failed to provide the correct user name or password. Tor is a software for enabling anonymous communication. It encrypts and randomly bounces communications through relays between a series of network nodes. The last Tor node is called the exit node. This can indicate unauthorized access to the RDS resources in your account, with the intent of hiding the anonymous user's true identity.

Remediation recommendations:

If this activity is unexpected for the associated database, it may indicate that there is an overly permissive access policy to the database or the database is publicly exposed. It is recommended to place the database in a private VPC, and limit the security group rules to allow traffic only from the necessary sources. For more information, see Remediating potentially compromised database with failed login events.

Discovery:RDS/TorIPCaller

A Tor exit node IP address probed an RDS database in your account, no authentication attempt was made.

Default severity: Medium

- **Feature:** RDS login activity monitoring

This finding informs you that a Tor exit node IP address probed an RDS database in your AWS environment, though no login attempt was made. This may indicate that a potentially malicious actor is attempting to scan for publicly accessible infrastructure. Tor is a software for enabling anonymous communication. It encrypts and randmonly bounces communications through relays between a series of network nodes. The last Tor node is called the exit node. This can indicate unauthorized access to the RDS resources in your account, with the intent of hiding the potentially malicious actor's true identity.

Remediation recommendations:

If this activity is unexpected for the associated database, it may indicate that there is an overly permissive access policy to the database or the database is publicly exposed. It is recommended to place the database in a private VPC, and limit the security group rules to allow traffic only from the necessary sources. For more information, see Remediating potentially compromised database with failed login events.

Lambda Protection finding types

This section describes the finding types that are specific to your AWS Lambda resources and have the `resourceType` listed as `Lambda`. For all Lambda findings, we recommend that you examine the resource in question and determine if it is behaving in an expected manner. If the activity is authorized, you can use Suppression rules or Trusted IP and threat lists to prevent false positive notifications for that resource.

If the activity is unexpected, the security best practice is to assume that Lambda has been potentially compromised and follow the remediation recommendations.

Topics

- Backdoor:Lambda/C&CActivity.B
- CryptoCurrency:Lambda/BitcoinTool.B
- Trojan:Lambda/BlackholeTraffic
- Trojan:Lambda/DropPoint
- UnauthorizedAccess:Lambda/MaliciousIPCaller.Custom
- UnauthorizedAccess:Lambda/TorClient
- UnauthorizedAccess:Lambda/TorRelay

Backdoor:Lambda/C&CActivity.B

A Lambda function is querying an IP address that is associated with a known command and control server.

Default severity: High

- **Feature:** Lambda Network Activity Monitoring

This finding informs you that a listed Lambda function within your AWS environment is querying an IP address that is associated with a known command and control (C&C) server. The Lambda function associated to the generated finding is potentially compromised. C&C servers are computers that issue commands to members of a botnet.

A botnet is a collection of internet-connected devices, which might include PCs, servers, mobile devices, and Internet of Things devices, that is infected and controlled by a common type of malware. Botnets are often used to distribute malware and gather misappropriated information, such as credit card numbers. Depending on the purpose and structure of the botnet, the C&C server might also issue commands to begin a distributed denial of service.

Remediation recommendations:

If this activity is unexpected, your Lambda function may be compromised. For more information, see Remediating a potentially compromised Lambda function.

CryptoCurrency:Lambda/BitcoinTool.B

A Lambda function is querying an IP address that is associated with a cryptocurrency-related activity.

Default severity: High

- **Feature:** Lambda Network Activity Monitoring

This finding informs you that the listed Lambda function in your AWS environment is querying an IP address that is associated with a Bitcoin or other cryptocurrency-related activity. Threat actors may seek to take control over Lambda functions in order to maliciously repurpose them for unauthorized cryptocurrency mining.

Remediation recommendations:

If you use this Lambda function to mine or manage cryptocurrency, or this function is otherwise involved in a blockchain activity, it is potentially an expected activity for your environment. If this is the case in your AWS environment, we recommend that you set up a suppression rule for this finding. The suppression rule should consist of two filter criteria. The first criterion should use the finding type attribute with a value of CryptoCurrency:Lambda/BitcoinTool.B. The second filter criterion should be the Lambda function name of the function involved in blockchain activity. For information about creating suppression rules, see Suppression rules.

If this activity is unexpected, your Lambda function is potentially compromised. For more information, see Remediating a potentially compromised Lambda function.

Trojan:Lambda/BlackholeTraffic

A Lambda function is attempting to communicate with an IP address of a remote host that is a known black hole.

Default severity: Medium

- **Feature:** Lambda Network Activity Monitoring

This finding informs you that a listed Lambda function within your AWS environment is trying to communicate with an IP address of a black hole (or a sink hole). Black holes are places in the network where incoming or outgoing traffic is silently discarded without informing the source that the data didn't reach its intended recipient. A black hole IP address specifies a host machine that is not running or an address to which no host has been assigned. The listed Lambda function is potentially compromised.

Remediation recommendations:

If this activity is unexpected, your Lambda function may be compromised. For more information, see Remediating a potentially compromised Lambda function.

Trojan:Lambda/DropPoint

A Lambda function is attempting to communicate with an IP address of a remote host that is known to hold credentials and other stolen data captured by malware.

Default severity: Medium

- **Feature:** Lambda Network Activity Monitoring

This finding informs you that a listed Lambda function within your AWS environment is trying to communicate with an IP address of a remote host that is known to hold credentials and other stolen data captured by malware.

Remediation recommendations:

If this activity is unexpected, your Lambda function may be compromised. For more information, see Remediating a potentially compromised Lambda function.

UnauthorizedAccess:Lambda/MaliciousIPCaller.Custom

A Lambda function is making connections to an IP address on a custom threat list.

Default severity: Medium

- **Feature:** Lambda Network Activity Monitoring

This finding informs you that a Lambda function in your AWS environment is communicating with an IP address included on a threat list that you uploaded. In GuardDuty, a threat list consists of known malicious IP addresses. GuardDuty generates findings based on the uploaded threat lists. You can view the details of the threat list in the finding details on the GuardDuty console.

Remediation recommendations:

If this activity is unexpected, your Lambda function may be compromised. For more information, see Remediating a potentially compromised Lambda function.

UnauthorizedAccess:Lambda/TorClient

A Lambda function is making connections to a Tor Guard or an Authority node.

Default severity: High

- **Feature:** Lambda Network Activity Monitoring

This finding informs you that a Lambda function in your AWS environment is making connections to a Tor Guard or an Authority node. Tor is software for enabling anonymous communication. Tor Guards and Authority node act as initial gateways into a Tor network. This traffic can indicate that this Lambda function has been potentially compromised. It is now acting as a client on a Tor network.

Remediation recommendations:

If this activity is unexpected, your Lambda function may be compromised. For more information, see Remediating a potentially compromised Lambda function.

UnauthorizedAccess:Lambda/TorRelay

A Lambda function is making connections to a Tor network as a Tor relay.

Default severity: High

- **Feature:** Lambda Network Activity Monitoring

This finding informs you that a Lambda function in your AWS environment is making connections to a Tor network in a manner that suggests that it's acting as a Tor relay. Tor is software for enabling anonymous communication. Tor enables anonymous communication by forwarding the client's potentially illicit traffic from one Tor relay to another.

Remediation recommendations:

If this activity is unexpected, your Lambda function may be compromised. For more information, see Remediating a potentially compromised Lambda function.

Retired finding types

A finding is a notification that contains details about a potential security issue that GuardDuty discovers. For information about important changes to the GuardDuty finding types, including newly added or retired finding types, see Document history for Amazon GuardDuty.

The following finding types are retired and no longer generated by GuardDuty.

Important

You can't reactivate retired GuardDuty finding types.

Topics

- Exfiltration:S3/ObjectRead.Unusual
- Impact:S3/PermissionsModification.Unusual
- Impact:S3/ObjectDelete.Unusual
- Discovery:S3/BucketEnumeration.Unusual
- Persistence:IAMUser/NetworkPermissions
- Persistence:IAMUser/ResourcePermissions
- Persistence:IAMUser/UserPermissions
- PrivilegeEscalation:IAMUser/AdministrativePermissions
- Recon:IAMUser/NetworkPermissions
- Recon:IAMUser/ResourcePermissions
- Recon:IAMUser/UserPermissions
- ResourceConsumption:IAMUser/ComputeResources
- Stealth:IAMUser/LoggingConfigurationModified
- UnauthorizedAccess:IAMUser/ConsoleLogin
- UnauthorizedAccess:EC2/TorIPCaller
- Backdoor:EC2/XORDDOS
- Behavior:IAMUser/InstanceLaunchUnusual
- CryptoCurrency:EC2/BitcoinTool.A
- UnauthorizedAccess:IAMUser/UnusualASNCaller

Exfiltration:S3/ObjectRead.Unusual

An IAM entity invoked an S3 API in a suspicious way.

Default severity: Medium*

Note

This finding's default severity is Medium. However, if the API is invoked using temporary AWS credentials that are created on an instance, the finding's severity is High.

- **Data source:** CloudTrail data events for S3

This finding informs you that a IAM entity in your AWS environment is making API calls that involve an S3 bucket and that differ from that entity's established baseline. The API call used in this activity is associated with the exfiltration stage of an attack, wherein and attacker is attempting to collect data. This activity is suspicious because the way the IAM entity invoked the API was unusual. For example, this IAM entity had no prior history of invoking this type of API, or the API was invoked from an unusual location.

Remediation recommendations:

If this activity is unexpected for the associated principal, it may indicate that the credentials have been exposed or your S3 permissions are not restrictive enough. For more information, see Remediating a potentially compromised S3 bucket.

Impact:S3/PermissionsModification.Unusual

An IAM entity invoked an API to modify permissions on one or more S3 resources.

Default severity: Medium*

Note

This finding's default severity is Medium. However, if the API is invoked using temporary AWS credentials that are created on an instance, the finding's severity is High.

This finding informs you that an IAM entity is making API calls designed to modify the permissions on one or more buckets or objects in your AWS environment. This action may be performed by an attacker to allow information to be shared outside of the account. This activity is suspicious because the way the IAM entity invoked the API was unusual. For example, this IAM entity had no prior history of invoking this type of API, or the API was invoked from an unusual location.

Remediation recommendations:

If this activity is unexpected for the associated principal, it may indicate that the credentials have been exposed or your S3 permissions are not restrictive enough. For more information, see Remediating a potentially compromised S3 bucket.

Impact:S3/ObjectDelete.Unusual

An IAM entity invoked an API used to delete data in an S3 bucket.

Default severity: Medium*

> **Note**
>
> This finding's default severity is Medium. However, if the API is invoked using temporary AWS credentials that are created on an instance, the finding's severity is High.

This finding informs you that a specific IAM entity in your AWS environment is making API calls designed to delete data in the listed S3 bucket by deleting the bucket itself. This activity is suspicious because the way the IAM entity invoked the API was unusual. For example, this IAM entity had no prior history of invoking this type of API, or the API was invoked from an unusual location.

Remediation recommendations:

If this activity is unexpected for the associated principal, it may indicate that the credentials have been exposed or your S3 permissions are not restrictive enough. For more information, see Remediating a potentially compromised S3 bucket.

Discovery:S3/BucketEnumeration.Unusual

An IAM entity invoked an S3 API used to discover S3 buckets within your network.

Default severity: Medium*

> **Note**
>
> This finding's default severity is Medium. However, if the API is invoked using temporary AWS credentials that are created on an instance, the finding's severity is High.

This finding informs you that an IAM entity has invoked an S3 API to discover S3 buckets in your environment, such as `ListBuckets`. This type of activity is associated with the discovery stage of an attack wherein an attacker is gathering information to determine if your AWS environment is susceptible to a broader attack. This activity is suspicious because the way the IAM entity invoked the API was unusual. For example, this IAM entity had no prior history of invoking this type of API, or the API was invoked from an unusual location.

Remediation recommendations:

If this activity is unexpected for the associated principal, it may indicate that the credentials have been exposed or your S3 permissions are not restrictive enough. For more information, see [Remediating a potentially compromised S3 bucket](#).

Persistence:IAMUser/NetworkPermissions

An IAM entity invoked an API commonly used to change the network access permissions for security groups, routes, and ACLs in your AWS account.

Default severity: Medium*

> **Note**
>
> This finding's default severity is Medium. However, if the API is invoked using temporary AWS credentials that are created on an instance, the finding's severity is High.

This finding indicates that a specific principal (AWS account root user, IAM role, or user) in your AWS environment is exhibiting behavior that is different from the established baseline. This principal has no prior history of invoking this API.

This finding is triggered when network configuration settings are changed under suspicious circumstances, such as when a principal invokes the `CreateSecurityGroup` API with no prior history of doing so. Attackers often attempt to change security groups to allow certain inbound traffic on various ports to improve their ability to access an EC2 instance.

Remediation recommendations:

If this activity is unexpected, your credentials may be compromised. For more information, see Remediating potentially compromised AWS credentials.

Persistence:IAMUser/ResourcePermissions

A principal invoked an API commonly used to change the security access policies of various resources in your AWS account.

Default severity: Medium*

> **Note**
>
> This finding's default severity is Medium. However, if the API is invoked is using temporary AWS credentials that are created on an instance, the finding's severity is High.

This finding indicates that a specific principal (AWS account root user, IAM role, or user) in your AWS environment is exhibiting behavior that is different from the established baseline. This principal has no prior history of invoking this API.

This finding is triggered when a change is detected to policies or permissions attached to AWS resources, such as when a principal in your AWS environment invokes the `PutBucketPolicy` API with no prior history of doing so. Some services, such as Amazon S3, support resource-attached permissions that grant one or more principals access to the resource. With stolen credentials, attackers can change the policies attached to a resource in order to gain access to that resource.

Remediation recommendations:

If this activity is unexpected, your credentials may be compromised. For more information, see Remediating potentially compromised AWS credentials.

Persistence:IAMUser/UserPermissions

A principal invoked an API commonly used to add, modify, or delete IAM users, groups or policies in your AWS account.

Default severity: Medium*

> **Note**
>
> This finding's default severity is Medium. However, if the API is invoked using temporary AWS credentials that are created on an instance, the finding's severity is High.

This finding indicates that a specific principal (AWS account root user, IAM role, or user) in your AWS environment is exhibiting behavior that is different from the established baseline. This principal has no prior history of invoking this API.

This finding is triggered by suspicious changes to the user-related permissions in your AWS environment, such as when a principal in your AWS environment invokes the `AttachUserPolicy` API with no prior history of doing so. Attackers may use stolen credentials to create new users, add access policies to existing users, or create access keys to maximize their access to an account, even if their original access point is closed. For example, the owner of the account might notice that a particular IAM user or password was stolen and delete it from the account. However, they might not delete other users that were created by a fraudulently created admin principal, leaving their AWS account accessible to the attacker.

Remediation recommendations:

If this activity is unexpected, your credentials may be compromised. For more information, see Remediating potentially compromised AWS credentials.

PrivilegeEscalation:IAMUser/AdministrativePermissions

A principal has attempted to assign a highly permissive policy to themselves.

Default severity: Low*

> **Note**
>
> This finding's severity is Low if the attempt at privilege escalation was unsuccessful, and Medium if the attempt at privilege escalation was successful.

This finding indicates that a specific IAM entity in your AWS environment is exhibiting behavior that can be indicative of a privilege escalation attack. This finding is triggered when an IAM user or role attempts to assign a highly permissive policy to themselves. If the user or role in question is not meant to have administrative privileges, either the user's credentials may be compromised or the role's permissions may not be configured properly.

Attackers will use stolen credentials to create new users, add access policies to existing users, or create access keys to maximize their access to an account even if their original access point is closed. For example, the owner of the account might notice that a particular IAM user's sign-in credential was stolen and deleted it from the account, but might not delete other users that were created by a fraudulently created admin principal, leaving their AWS account still accessible to the attacker.

Remediation recommendations:

If this activity is unexpected, your credentials may be compromised. For more information, see [Remediating potentially compromised AWS credentials](#).

Recon:IAMUser/NetworkPermissions

A principal invoked an API commonly used to change the network access permissions for security groups, routes, and ACLs in your AWS account.

Default severity: Medium*

> **Note**
>
> This finding's default severity is Medium. However, if the API is invoked using temporary AWS credentials that are created on an instance, the finding's severity is High.

This finding indicates that a specific principal (AWS account root user, IAM role, or user) in your AWS environment is exhibiting behavior that is different from the established baseline. This principal has no prior history of invoking this API.

This finding is triggered when resource access permissions in your AWS account are probed under suspicious circumstances. For example, if a principal invoked the `DescribeInstances` API with no prior history of doing so. An attacker might use stolen credentials to perform this type of reconnaissance of your AWS resources in order to find more valuable credentials or determine the capabilities of the credentials they already have.

Remediation recommendations:

If this activity is unexpected, your credentials may be compromised. For more information, see [Remediating potentially compromised AWS credentials].

Recon:IAMUser/ResourcePermissions

A principal invoked an API commonly used to change the security access policies of various resources in your AWS account.

Default severity: Medium*

> **Note**
>
> This finding's default severity is Medium. However, if the API is invoked using temporary AWS credentials that are created on an instance, the finding's severity is High.

This finding indicates that a specific principal (AWS account root user, IAM role, or user) in your AWS environment is exhibiting behavior that is different from the established baseline. This principal has no prior history of invoking this API.

This finding is triggered when resource access permissions in your AWS account are probed under suspicious circumstances. For example, if a principal invoked the `DescribeInstances` API with no prior history of doing so. An attacker might use stolen credentials to perform this type of reconnaissance of your AWS resources in order to find more valuable credentials or determine the capabilities of the credentials they already have.

Remediation recommendations:

If this activity is unexpected, your credentials may be compromised. For more information, see [Remediating potentially compromised AWS credentials].

Recon:IAMUser/UserPermissions

A principal invoked an API commonly used to add, modify, or delete IAM users, groups or policies in your AWS account.

Default severity: Medium*

> **Note**
>
> This finding's default severity is Medium. However, if the API is invoked using temporary AWS credentials that are created on an instance, the finding's severity is High.

This finding is triggered when user permissions in your AWS environment are probed under suspicious circumstances. For example, if a principal (AWS account root user, IAM role, or IAM user) invoked the `ListInstanceProfilesForRole` API with no prior history of doing so. An attacker might use stolen credentials to perform this type of reconnaissance of your AWS resources in order to find more valuable credentials or determine the capabilities of the credentials they already have.

This finding indicates that a specific principal in your AWS environment is exhibiting behavior that is different from the established baseline. This principal has no prior history of invoking this API in this way.

Remediation recommendations:

If this activity is unexpected, your credentials may be compromised. For more information, see Remediating potentially compromised AWS credentials.

ResourceConsumption:IAMUser/ComputeResources

A principal invoked an API commonly used to launch Compute resources like EC2 Instances.

Default severity: Medium*

> **Note**
>
> This finding's default severity is Medium. However, if the API is invoked using temporary AWS credentials that are created on an instance, the finding's severity is High.

This finding is triggered when EC2 instances in the listed account within your AWS environment are launched under suspicious circumstances. This finding indicates that a specific principal in your AWS environment is exhibiting behavior that is different from the established baseline; for example, if a principal (AWS account root user, IAM role, or IAM user) invoked the `RunInstances` API with no prior history of doing so. This might be an indication of an attacker using stolen credentials to steal compute time (possibly for cryptocurrency mining or password cracking). It can also be an indication of an attacker using an EC2 instance in your AWS environment and its credentials to maintain access to your account.

Remediation recommendations:

If this activity is unexpected, your credentials may be compromised. For more information, see Remediating potentially compromised AWS credentials.

Stealth:IAMUser/LoggingConfigurationModified

A principal invoked an API commonly used to stop CloudTrail Logging, delete existing logs, and otherwise eliminate traces of activity in your AWS account.

Default severity: Medium*

> **Note**
>
> This finding's default severity is Medium. However, if the API is invoked using temporary AWS credentials that are created on an instance, the finding's severity is High.

This finding is triggered when the logging configuration in the listed AWS account within your environment is modified under suspicious circumstances. This finding informs you that a specific principal in your AWS environment is exhibiting behavior that is different from the established baseline; for example, if a principal (AWS account root user, IAM role, or IAM user) invoked the `StopLogging` API with no prior history of doing so. This can be an indication of an attacker trying to cover their tracks by eliminating any trace of their activity.

Remediation recommendations:

If this activity is unexpected, your credentials may be compromised. For more information, see Remediating potentially compromised AWS credentials.

UnauthorizedAccess:IAMUser/ConsoleLogin

An unusual console login by a principal in your AWS account was observed.

Default severity: Medium*

> **Note**
>
> This finding's default severity is Medium. However, if the API is invoked using temporary AWS credentials that are created on an instance, the finding's severity is High.

This finding is triggered when a console login is detected under suspicious circumstances. For example, if a principal with no prior history of doing so, invoked the ConsoleLogin API from a never-before-used client or an unusual location. This could be an indication of stolen credentials being used to gain access to your AWS account, or a valid user accessing the account in an invalid or less secure manner (for example, not over an approved VPN).

This finding informs you that a specific principal in your AWS environment is exhibiting behavior that is different from the established baseline. This principal has no prior history of login activity using this client application from this specific location.

Remediation recommendations:

If this activity is unexpected, your credentials may be compromised. For more information, see Remediating potentially compromised AWS credentials.

UnauthorizedAccess:EC2/TorIPCaller

Your EC2 instance is receiving inbound connections from a Tor exit node.

Default severity: Medium

This finding informs you that an EC2 instance in your AWS environment is receiving inbound connections from a Tor exit node. Tor is software for enabling anonymous communication. It encrypts and randomly bounces communications through relays between a series of network nodes. The last Tor node is called the exit node. This finding can indicate unauthorized access to your AWS resources with the intent of hiding the attacker's true identity.

Remediation recommendations:

If this activity is unexpected, your instance may be compromised. For more information, see Remediating a potentially compromised Amazon EC2 instance.

Backdoor:EC2/XORDDOS

An EC2 instance is attempting to communicate with an IP address that is associated with XOR DDoS malware.

Default severity: High

This finding informs you that an EC2 instance in your AWS environment is attempting to communicate with an IP address that is associated with XOR DDoS malware. This EC2 instance might be compromised. XOR DDoS is Trojan malware that hijacks Linux systems. To gain access to the system, it launches a brute force attack in order to discover the password to Secure Shell (SSH) services on Linux. After SSH credentials are acquired and the login is successful, it uses root user privileges to run a script that downloads and installs XOR DDoS. This malware is then used as part of a botnet to launch distributed denial of service (DDoS) attacks against other targets.

Remediation recommendations:

If this activity is unexpected, your instance may be compromised. For more information, see Remediating a potentially compromised Amazon EC2 instance.

Behavior:IAMUser/InstanceLaunchUnusual

A user launched an EC2 instance of an unusual type.

Default severity: High

This finding informs you that a specific user in your AWS environment is exhibiting behavior that is different from the established baseline. This user has no prior history of launching an EC2 instance of this type. Your sign-in credentials might be compromised.

Remediation recommendations:

If this activity is unexpected, your credentials may be compromised. For more information, see Remediating potentially compromised AWS credentials.

CryptoCurrency:EC2/BitcoinTool.A

EC2 instance is communicating with Bitcoin mining pools.

Default severity: High

This finding informs you that an EC2 instance in your AWS environment is communicating with Bitcoin mining pools. In the field of cryptocurrency mining, a mining pool is the pooling of resources by miners who share their processing power over a network to split the reward according to the amount of work they contributed to solving a block. Unless you use this EC2 instance for Bitcoin mining, your EC2 instance might be compromised.

Remediation recommendations:

If this activity is unexpected, your instance may be compromised. For more information, see Remediating a potentially compromised Amazon EC2 instance.

UnauthorizedAccess:IAMUser/UnusualASNCaller

An API was invoked from an IP address of an unusual network.

Default severity: High

This finding informs you that certain activity was invoked from an IP address of an unusual network. This network was never observed throughout the AWS usage history of the described user. This activity can include a console login, an attempt to launch an EC2 instance, create a new IAM user, modify your AWS privileges, etc. This can indicate unauthorized access to your AWS resources.

Remediation recommendations:

If this activity is unexpected, your credentials may be compromised. For more information, see Remediating potentially compromised AWS credentials.

GuardDuty finding types by potentially impacted resources

The following pages are categorized by the potentially impacted resource type associated to a GuardDuty finding:

- EC2 finding types
- IAM finding types
- Attack sequence finding types
- S3 Protection finding types
- EKS Protection finding types
- Runtime Monitoring finding types

- Malware Protection for EC2 finding types
- Malware Protection for S3 finding type
- RDS Protection finding types
- Lambda Protection finding types

GuardDuty active finding types

The following table shows all of the active finding types sorted by the foundational data source or feature, as applicable. In the following table, some of the findings have their *Finding severity* column values marked with an asterisk (*) or a plus sign (+):

*These finding types have variable severity. A finding of a particular type may have a different severity depending on the context specific to the finding. For more information about a finding type, view its detailed description.

+EC2 findings that use VPC flow logs as a data source do not support IPv6 traffic.

Finding type	Resource type	Foundational data source/Feature	Finding severity
Discovery:S3/AnomalousBehavior	Amazon S3	CloudTrail data events for S3	Low
Discovery:S3/MaliciousIPCaller	Amazon S3	CloudTrail data events for S3	High
Discovery:S3/MaliciousIPCaller.Custom	Amazon S3	CloudTrail data events for S3	High
Discovery:S3/TorIPCaller	Amazon S3	CloudTrail data events for S3	Medium
Exfiltration:S3/AnomalousBehavior	Amazon S3	CloudTrail data events for S3	High
Exfiltration:S3/MaliciousIPCaller	Amazon S3	CloudTrail data events for S3	High

Finding type	Resource type	Foundational data source/Feature	Finding severity
Impact:EC2/MaliciousDomainRequest.Custom	Amazon EC2	DNS logs	Medium
Impact:S3/AnomalousBehavior.Delete	Amazon S3	CloudTrail data events for S3	High
Impact:S3/AnomalousBehavior.Permission	Amazon S3	CloudTrail data events for S3	High
Impact:S3/AnomalousBehavior.Write	Amazon S3	CloudTrail data events for S3	Medium
Impact:S3/MaliciousIPCaller	Amazon S3	CloudTrail data events for S3	High
PenTest:S3/KaliLinux	Amazon S3	CloudTrail data events for S3	Medium
PenTest:S3/ParrotLinux	Amazon S3	CloudTrail data events for S3	Medium
PenTest:S3/PentooLinux	Amazon S3	CloudTrail data events for S3	Medium
UnauthorizedAccess:S3/TorIPCaller	Amazon S3	CloudTrail data events for S3	High
UnauthorizedAccess:S3/MaliciousIPCaller.Custom	Amazon S3	CloudTrail data events for S3	High
CredentialAccess:IAMUser/AnomalousBehavior	IAM	CloudTrail management events	Medium

Finding type	Resource type	Foundational data source/Feature	Finding severity
DefenseEvasion:IAMUser/AnomalousBehavior	IAM	CloudTrail management events	Medium
Discovery:IAMUser/AnomalousBehavior	IAM	CloudTrail management events	Low
Exfiltration:IAMUser/AnomalousBehavior	IAM	CloudTrail management events	High
Impact:IAMUser/AnomalousBehavior	IAM	CloudTrail management events	High
InitialAccess:IAMUser/AnomalousBehavior	IAM	CloudTrail management events	Medium
PenTest:IAMUser/KaliLinux	IAM	CloudTrail management events	Medium
PenTest:IAMUser/ParrotLinux	IAM	CloudTrail management events	Medium
PenTest:IAMUser/PentooLinux	IAM	CloudTrail management events	Medium
Persistence:IAMUser/AnomalousBehavior	IAM	CloudTrail management events	Medium
Stealth:IAMUser/PasswordPolicyChange	IAM	CloudTrail management events	Low*
UnauthorizedAccess:IAMUser/InstanceCredentialExfiltration.InsideAWS	IAM	CloudTrail management events	High*

Finding type	Resource type	Foundational data source/Feature	Finding severity
Policy:S3/AccountBlockPublicAccessDisabled	Amazon S3	CloudTrail management events	Low
Policy:S3/BucketAnonymousAccessGranted	Amazon S3	CloudTrail management events	High
Policy:S3/BucketBlockPublicAccessDisabled	Amazon S3	CloudTrail management events	Low
Policy:S3/BucketPublicAccessGranted	Amazon S3	CloudTrail management events	High
PrivilegeEscalation:IAMUser/AnomalousBehavior	IAM	CloudTrail management events	Medium
Recon:IAMUser/MaliciousIPCaller	IAM	CloudTrail management events	Medium
Recon:IAMUser/MaliciousIPCaller.Custom	IAM	CloudTrail management events	Medium
Recon:IAMUser/TorIPCaller	IAM	CloudTrail management events	Medium
Stealth:IAMUser/CloudTrailLoggingDisabled	IAM	CloudTrail management events	Low
Stealth:S3/ServerAccessLoggingDisabled	Amazon S3	CloudTrail management events	Low

Finding type	Resource type	Foundational data source/Feature	Finding severity
UnauthorizedAccess:IAMUser/ConsoleLoginSuccess.B	IAM	CloudTrail management events	Medium
UnauthorizedAccess:IAMUser/MaliciousIPCaller	IAM	CloudTrail management events	Medium
UnauthorizedAccess:IAMUser/MaliciousIPCaller.Custom	IAM	CloudTrail management events	Medium
UnauthorizedAccess:IAMUser/TorIPCaller	IAM	CloudTrail management events	Medium
Policy:IAMUser/RootCredentialUsage	IAM	CloudTrail management events or CloudTrail data events for S3	Low
Policy:IAMUser/ShortTermRootCredentialUsage	IAM	CloudTrail management events or CloudTrail data events for S3	Low
UnauthorizedAccess:IAMUser/InstanceCredentialExfiltration.OutsideAWS	IAM	CloudTrail management events or CloudTrail data events for S3	High

Finding type	Resource type	Foundational data source/Feature	Finding severity
AttackSequence:EKS/CompromisedCluster	Resources involved in attack sequence	• EKS audit log events • Runtime Monitoring for Amazon EKS • Amazon EKS malware detection for Amazon EC2 • AWS CloudTrail data events for S3 • AWS CloudTrail management events • VPC Flow Logs • Route53 Resolver DNS query logs	Critical
AttackSequence:IAM/CompromisedCredentials	Resources involved in attack sequence	CloudTrail management events	Critical
AttackSequence:S3/CompromisedData	Resources involved in attack sequence	CloudTrail management events and CloudTrail data events for S3	Critical
Backdoor:EC2/C&CActivity.B!DNS	Amazon EC2	DNS logs	High
CryptoCurrency:EC2/BitcoinTool.B!DNS	Amazon EC2	DNS logs	High

Finding type	Resource type	Foundational data source/Feature	Finding severity
Impact:EC2/AbusedDomainRequest.Reputation	Amazon EC2	DNS logs	Medium
Impact:EC2/BitcoinDomainRequest.Reputation	Amazon EC2	DNS logs	High
Impact:EC2/MaliciousDomainRequest.Reputation	Amazon EC2	DNS logs	High
Impact:EC2/SuspiciousDomainRequest.Reputation	Amazon EC2	DNS logs	Low
Trojan:EC2/BlackholeTraffic!DNS	Amazon EC2	DNS logs	Medium
Trojan:EC2/DGADomainRequest.B	Amazon EC2	DNS logs	High
Trojan:EC2/DGADomainRequest.C!DNS	Amazon EC2	DNS logs	High
Trojan:EC2/DNSDataExfiltration	Amazon EC2	DNS logs	High
Trojan:EC2/DriveBySourceTraffic!DNS	Amazon EC2	DNS logs	High
Trojan:EC2/DropPoint!DNS	Amazon EC2	DNS logs	Medium

Finding type	Resource type	Foundational data source/Feature	Finding severity
Trojan:EC2/PhishingDomainRequest!DNS	Amazon EC2	DNS logs	High
UnauthorizedAccess:EC2/MetadataDNSRebind	Amazon EC2	DNS logs	High
Execution:Container/MaliciousFile	Container	EBS Malware Protection	Varies depending on the detected threat
Execution:Container/SuspiciousFile	Container	EBS Malware Protection	Varies depending on the detected threat
Execution:EC2/MaliciousFile	Amazon EC2	EBS Malware Protection	Varies depending on the detected threat
Execution:EC2/SuspiciousFile	Amazon EC2	EBS Malware Protection	Varies depending on the detected threat
Execution:ECS/MaliciousFile	ECS	EBS Malware Protection	Varies depending on the detected threat
Execution:ECS/SuspiciousFile	ECS	EBS Malware Protection	Varies depending on the detected threat
Execution:Kubernetes/MaliciousFile	Kubernetes	EBS Malware Protection	Varies depending on the detected threat
Execution:Kubernetes/SuspiciousFile	Kubernetes	EBS Malware Protection	Varies depending on the detected threat
CredentialAccess:Kubernetes/AnomalousBehavior.SecretsAccessed	Kubernetes	EKS audit logs	Medium

Finding type	Resource type	Foundational data source/Feature	Finding severity
CredentialAccess:Kubernetes/MaliciousIPCaller	Kubernetes	EKS audit logs	High
CredentialAccess:Kubernetes/MaliciousIPCaller.Custom	Kubernetes	EKS audit logs	High
CredentialAccess:Kubernetes/SuccessfulAnonymousAccess	Kubernetes	EKS audit logs	High
CredentialAccess:Kubernetes/TorIPCaller	Kubernetes	EKS audit logs	High
DefenseEvasion:Kubernetes/MaliciousIPCaller	Kubernetes	EKS audit logs	High
DefenseEvasion:Kubernetes/MaliciousIPCaller.Custom	Kubernetes	EKS audit logs	High
DefenseEvasion:Kubernetes/SuccessfulAnonymousAccess	Kubernetes	EKS audit logs	High
DefenseEvasion:Kubernetes/TorIPCaller	Kubernetes	EKS audit logs	High
Discovery:Kubernetes/AnomalousBehavior.PermissionChecked	Kubernetes	EKS audit logs	Low
Discovery:Kubernetes/MaliciousIPCaller	Kubernetes	EKS audit logs	Medium

Finding type	Resource type	Foundational data source/Feature	Finding severity
Discovery:Kubernetes/MaliciousIPCaller.Custom	Kubernetes	EKS audit logs	Medium
Discovery:Kubernetes/SuccessfulAnonymousAccess	Kubernetes	EKS audit logs	Medium
Discovery:Kubernetes/TorIPCaller	Kubernetes	EKS audit logs	Medium
Execution:Kubernetes/ExecInKubeSystemPod	Kubernetes	EKS audit logs	Medium
Execution:Kubernetes/AnomalousBehavior.ExecInPod	Kubernetes	EKS audit logs	Medium
Execution:Kubernetes/AnomalousBehavior.WorkloadDeployed	Kubernetes	EKS audit logs	Low
Impact:Kubernetes/MaliciousIPCaller	Kubernetes	EKS audit logs	High
Impact:Kubernetes/MaliciousIPCaller.Custom	Kubernetes	EKS audit logs	High
Impact:Kubernetes/SuccessfulAnonymousAccess	Kubernetes	EKS audit logs	High
Impact:Kubernetes/TorIPCaller	Kubernetes	EKS audit logs	High

Finding type	Resource type	Foundational data source/Feature	Finding severity
Persistence:Kubernetes/ContainerWithSensitiveMount	Kubernetes	EKS audit logs	Medium
Persistence:Kubernetes/MaliciousIPCaller	Kubernetes	EKS audit logs	Medium
Persistence:Kubernetes/MaliciousIPCaller.Custom	Kubernetes	EKS audit logs	Medium
Persistence:Kubernetes/SuccessfulAnonymousAccess	Kubernetes	EKS audit logs	High
Persistence:Kubernetes/TorIPCaller	Kubernetes	EKS audit logs	Medium
Policy:Kubernetes/AdminAccessToDefaultServiceAccount	Kubernetes	EKS audit logs	High
Policy:Kubernetes/AnonymousAccessGranted	Kubernetes	EKS audit logs	High
Policy:Kubernetes/KubeflowDashboardExposed	Kubernetes	EKS audit logs	Medium
Policy:Kubernetes/ExposedDashboard	Kubernetes	EKS audit logs	Medium

Finding type	Resource type	Foundational data source/Feature	Finding severity
PrivilegeEscalation:Kubernetes/AnomalousBehavior.RoleBindingCreated	Kubernetes	EKS audit logs	Medium*
PrivilegeEscalation:Kubernetes/AnomalousBehavior.RoleCreated	Kubernetes	EKS audit logs	Low
Persistence:Kubernetes/AnomalousBehavior.WorkloadDeployed!ContainerWithSensitiveMount	Kubernetes	EKS audit logs	High
PrivilegeEscalation:Kubernetes/AnomalousBehavior.WorkloadDeployed!PrivilegedContainer	Kubernetes	EKS audit logs	High
PrivilegeEscalation:Kubernetes/PrivilegedContainer	Kubernetes	EKS audit logs	Medium
Backdoor:Lambda/C&CActivity.B	Lambda	Lambda Network Activity Monitoring	High
CryptoCurrency:Lambda/BitcoinTool.B	Lambda	Lambda Network Activity Monitoring	High
Trojan:Lambda/BlackholeTraffic	Lambda	Lambda Network Activity Monitoring	Medium

Finding type	Resource type	Foundational data source/Feature	Finding severity
Trojan:Lambda/DropPoint	Lambda	Lambda Network Activity Monitoring	Medium
UnauthorizedAccess:Lambda/MaliciousIPCaller.Custom	Lambda	Lambda Network Activity Monitoring	Medium
UnauthorizedAccess:Lambda/TorClient	Lambda	Lambda Network Activity Monitoring	High
UnauthorizedAccess:Lambda/TorRelay	Lambda	Lambda Network Activity Monitoring	High
Object:S3/MaliciousFile	S3Object	Malware Protection for S3	High
CredentialAccess:RDS/AnomalousBehavior.FailedLogin	Supported Amazon Aurora, Amazon RDS, and Aurora Limitless databases	RDS Login Activity Monitoring	Low
CredentialAccess:RDS/AnomalousBehavior.SuccessfulBruteForce	Supported Amazon Aurora, Amazon RDS, and Aurora Limitless databases	RDS Login Activity Monitoring	High
CredentialAccess:RDS/AnomalousBehavior.SuccessfulLogin	Supported Amazon Aurora, Amazon RDS, and Aurora Limitless databases	RDS Login Activity Monitoring	Variable*
CredentialAccess:RDS/MaliciousIPCaller.FailedLogin	Supported Amazon Aurora, Amazon RDS, and Aurora Limitless databases	RDS Login Activity Monitoring	Medium

Finding type	Resource type	Foundational data source/Feature	Finding severity
CredentialAccess:RDS/MaliciousIPCaller.SuccessfulLogin	Supported Amazon Aurora, Amazon RDS, and Aurora Limitless databases	RDS Login Activity Monitoring	High
CredentialAccess:RDS/TorIPCaller.FailedLogin	Supported Amazon Aurora, Amazon RDS, and Aurora Limitless databases	RDS Login Activity Monitoring	Medium
CredentialAccess:RDS/TorIPCaller.SuccessfulLogin	Supported Amazon Aurora, Amazon RDS, and Aurora Limitless databases	RDS Login Activity Monitoring	High
Discovery:RDS/MaliciousIPCaller	Supported Amazon Aurora, Amazon RDS, and Aurora Limitless databases	RDS Login Activity Monitoring	Medium
Discovery:RDS/TorIPCaller	Supported Amazon Aurora, Amazon RDS, and Aurora Limitless databases	RDS Login Activity Monitoring	Medium
Backdoor:Runtime/C&CActivity.B	Instance, EKS cluster, ECS cluster, or container	Runtime Monitoring	High
Backdoor:Runtime/C&CActivity.B!DNS	Instance, EKS cluster, ECS cluster, or container	Runtime Monitoring	High

Finding type	Resource type	Foundational data source/Feature	Finding severity
CryptoCurrency:Runtime/BitcoinTool.B	Instance, EKS cluster, ECS cluster, or container	Runtime Monitoring	High
CryptoCurrency:Runtime/BitcoinTool.B!DNS	Instance, EKS cluster, ECS cluster, or container	Runtime Monitoring	High
DefenseEvasion:Runtime/FilelessExecution	Instance, EKS cluster, ECS cluster, or container	Runtime Monitoring	Medium
DefenseEvasion:Runtime/ProcessInjection.Proc	Instance, EKS cluster, ECS cluster, or container	Runtime Monitoring	High
DefenseEvasion:Runtime/ProcessInjection.Ptrace	Instance, EKS cluster, ECS cluster, or container	Runtime Monitoring	Medium
DefenseEvasion:Runtime/ProcessInjection.VirtualMemoryWrite	Instance, EKS cluster, ECS cluster, or container	Runtime Monitoring	High
DefenseEvasion:Runtime/PtraceAntiDebugging	Instance, EKS cluster, ECS cluster, or container	Runtime Monitoring	Low
DefenseEvasion:Runtime/SuspiciousCommand	Instance, EKS cluster, ECS cluster, or container	Runtime Monitoring	High
Discovery:Runtime/SuspiciousCommand	Instance, EKS cluster, ECS cluster, or container	Runtime Monitoring	Low

Finding type	Resource type	Foundational data source/Feature	Finding severity
Execution:Runtime/MaliciousFileExecuted	Instance, EKS cluster, ECS cluster, or container	Runtime Monitoring	High
Execution:Runtime/NewBinaryExecuted	Instance, EKS cluster, ECS cluster, or container	Runtime Monitoring	Medium
Execution:Runtime/NewLibraryLoaded	Instance, EKS cluster, ECS cluster, or container	Runtime Monitoring	Medium
Execution:Runtime/SuspiciousCommand	Instance, EKS cluster, ECS cluster, or container	Runtime Monitoring	Variable
Execution:Runtime/SuspiciousShellCreated	Instance, EKS cluster, ECS cluster, or container	Runtime Monitoring	Low
Execution:Runtime/SuspiciousTool	Instance, EKS cluster, ECS cluster, or container	Runtime Monitoring	Variable
Execution:Runtime/ReverseShell	Instance, EKS cluster, ECS cluster, or container	Runtime Monitoring	High
Impact:Runtime/AbusedDomainRequest.Reputation	Instance, EKS cluster, ECS cluster, or container	Runtime Monitoring	Medium
Impact:Runtime/BitcoinDomainRequest.Reputation	Instance, EKS cluster, ECS cluster, or container	Runtime Monitoring	High

Finding type	Resource type	Foundational data source/Feature	Finding severity
Impact:Runtime/CryptoMinerExecuted	Instance, EKS cluster, ECS cluster, or container	Runtime Monitoring	High
Impact:Runtime/MaliciousDomainRequest.Reputation	Instance, EKS cluster, ECS cluster, or container	Runtime Monitoring	Medium
Impact:Runtime/SuspiciousDomainRequest.Reputation	Instance, EKS cluster, ECS cluster, or container	Runtime Monitoring	Low
Persistence:Runtime/SuspiciousCommand	Instance, EKS cluster, ECS cluster, or container	Runtime Monitoring	Medium
PrivilegeEscalation:Runtime/CGroupsReleaseAgentModified	Instance, EKS cluster, ECS cluster, or container	Runtime Monitoring	High
PrivilegeEscalation:Runtime/ContainerMountsHostDirectory	Instance, EKS cluster, ECS cluster, or container	Runtime Monitoring	Medium
PrivilegeEscalation:Runtime/DockerSocketAccessed	Instance, EKS cluster, ECS cluster, or container	Runtime Monitoring	Medium
PrivilegeEscalation:Runtime/ElevationToRoot	Instance, EKS cluster, ECS cluster, or container	Runtime Monitoring	Medium
PrivilegeEscalation:Runtime/RuncContainerEscape	Instance, EKS cluster, ECS cluster, or container	Runtime Monitoring	High

Finding type	Resource type	Foundational data source/Feature	Finding severity
PrivilegeEscalation:Runtime/SuspiciousCommand	Instance, EKS cluster, ECS cluster, or container	Runtime Monitoring	Medium
PrivilegeEscalation:Runtime/UserfaultfdUsage	Instance, EKS cluster, ECS cluster, or container	Runtime Monitoring	Medium
Trojan:Runtime/BlackholeTraffic	Instance, EKS cluster, ECS cluster, or container	Runtime Monitoring	Medium
Trojan:Runtime/BlackholeTraffic!DNS	Instance, EKS cluster, ECS cluster, or container	Runtime Monitoring	Medium
Trojan:Runtime/DropPoint	Instance, EKS cluster, ECS cluster, or container	Runtime Monitoring	Medium
Trojan:Runtime/DGADomainRequest.C!DNS	Instance, EKS cluster, ECS cluster, or container	Runtime Monitoring	High
Trojan:Runtime/DriveBySourceTraffic!DNS	Instance, EKS cluster, ECS cluster, or container	Runtime Monitoring	High
Trojan:Runtime/DropPoint!DNS	Instance, EKS cluster, ECS cluster, or container	Runtime Monitoring	Medium
Trojan:Runtime/PhishingDomainRequest!DNS	Instance, EKS cluster, ECS cluster, or container	Runtime Monitoring	High

Finding type	Resource type	Foundational data source/Feature	Finding severity
UnauthorizedAccess:Runtime/MetadataDNSRebind	Instance, EKS cluster, ECS cluster, or container	Runtime Monitoring	High
UnauthorizedAccess:Runtime/TorClient	Instance, EKS cluster, ECS cluster, or container	Runtime Monitoring	High
UnauthorizedAccess:Runtime/TorRelay	Instance, EKS cluster, ECS cluster, or container	Runtime Monitoring	High
Backdoor:EC2/C&CActivity.B	Amazon EC2	VPC flow logs[±]	High
Backdoor:EC2/DenialOfService.Dns	Amazon EC2	VPC flow logs[±]	High
Backdoor:EC2/DenialOfService.Tcp	Amazon EC2	VPC flow logs[±]	High
Backdoor:EC2/DenialOfService.Udp	Amazon EC2	VPC flow logs[±]	High
Backdoor:EC2/DenialOfService.UdpOnTcpPorts	Amazon EC2	VPC flow logs[±]	High
Backdoor:EC2/DenialOfService.UnusualProtocol	Amazon EC2	VPC flow logs[±]	High
Backdoor:EC2/Spambot	Amazon EC2	VPC flow logs[±]	Medium

Finding type	Resource type	Foundational data source/Feature	Finding severity
Behavior:EC2/NetworkPortUnusual	Amazon EC2	VPC flow logs+	Medium
Behavior:EC2/TrafficVolumeUnusual	Amazon EC2	VPC flow logs+	Medium
CryptoCurrency:EC2/BitcoinTool.B	Amazon EC2	VPC flow logs+	High
DefenseEvasion:EC2/UnusualDNSResolver	Amazon EC2	VPC flow logs+	Medium
DefenseEvasion:EC2/UnusualDoHActivity	Amazon EC2	VPC flow logs+	Medium
DefenseEvasion:EC2/UnusualDoTActivity	Amazon EC2	VPC flow logs+	Medium
Impact:EC2/PortSweep	Amazon EC2	VPC flow logs+	High
Impact:EC2/WinRMBruteForce	Amazon EC2	VPC flow logs+	Low*
Recon:EC2/PortProbeEMRUnprotectedPort	Amazon EC2	VPC flow logs+	High
Recon:EC2/PortProbeUnprotectedPort	Amazon EC2	VPC flow logs+	Low*
Recon:EC2/Portscan	Amazon EC2	VPC flow logs+	Medium
Trojan:EC2/BlackholeTraffic	Amazon EC2	VPC flow logs+	Medium

Finding type	Resource type	Foundational data source/Feature	Finding severity
Trojan:EC2/DropPoint	Amazon EC2	VPC flow logs+	Medium
UnauthorizedAccess:EC2/MaliciousIPCaller.Custom	Amazon EC2	VPC flow logs+	Medium
UnauthorizedAccess:EC2/RDPBruteForce	Amazon EC2	VPC flow logs+	Low*
UnauthorizedAccess:EC2/SSHBruteForce	Amazon EC2	VPC flow logs+	Low*
UnauthorizedAccess:EC2/TorClient	Amazon EC2	VPC flow logs+	High
UnauthorizedAccess:EC2/TorRelay	Amazon EC2	VPC flow logs+	High

Understanding and generating Amazon GuardDuty findings

A GuardDuty finding represents a potential security issue detected within AWS accounts, workloads, and data. GuardDuty generates a finding whenever it detects unexpected and potentially malicious activity in your AWS environment.

You can view and manage your GuardDuty findings on the **Findings** page in the GuardDuty console, or by using the AWS CLI or API operations. For information on how you can manage GuardDuty findings, see [Managing Amazon GuardDuty findings](#).

Topics:

[GuardDuty finding format](#)

Understand the format of GuardDuty finding types and different threat purposes that GuardDuty tracks.

[Sample findings](#)

Generate sample findings in the GuardDuty console, or by using GuardDuty API or AWS CLI commands. The generated sample findings include fictitious details to help you understand the finding details associated with each GuardDuty finding. These findings are marked with a prefix **[SAMPLE]**.

[Test GuardDuty findings in dedicated accounts](#)

You can test specific GuardDuty findings in your environment. Run `guardduty-tester` script in a dedicated non-production AWS account. For GuardDuty to detect and simulate findings, it will deploy certain resources in your environment. This experience is different than generating sample findings.

[Viewing generated findings in GuardDuty console](#)

Learn how to review the generated findings in the GuardDuty console.

[Severity levels of GuardDuty findings](#)

Each GuardDuty finding has an associated severity level that reflects the potential risk in your AWS environment. This section explains what each severity level signify.

Finding details

Learn about the details associated with GuardDuty findings that get generated in your account. This topic includes the details associated with foundational threat detection, Extended Threat Detection, and dedicated protection plans in GuardDuty.

GuardDuty finding aggregation

Learn how GuardDuty handles multiple occurrences of the same finding type. By aggregating detected same finding types, GuardDuty updates the original finding type with the latest details.

GuardDuty finding types

This section enlists GuardDuty finding types by the associated Foundational data sources or Mapped GuardDuty feature. To learn about each finding type, select that finding for further details, such as its description and potential steps to remediate the finding.

GuardDuty finding format

When GuardDuty detects suspicious or unexpected behavior in your AWS environment, it generates a finding. A finding is a notification that contains the details about a potential security issue that GuardDuty discovers. The Viewing generated findings in GuardDuty console include information about what happened, which AWS resources were involved in the suspicious activity, when this activity took place, and related information that may help you understand the root cause.

One of the most useful pieces of information in the finding details is a **finding type**. The purpose of the finding type is to provide a concise yet readable description of the potential security issue. For example, the GuardDuty *Recon:EC2/PortProbeUnprotectedPort* finding type quickly informs you that somewhere in your AWS environment, an EC2 instance has an unprotected port that a potential attacker is probing.

GuardDuty uses the following format for naming the various types of findings that it generates:

ThreatPurpose:ResourceTypeAffected/ThreatFamilyName.DetectionMechanism!Artifact

Each part of this format represents an aspect of a finding type. These aspects have the following explanations:

- **ThreatPurpose** - describes the primary purpose of a threat, an attack type or a stage of a potential attack. See the following section for a complete list of GuardDuty threat purposes.

- **ResourceTypeAffected** - describes which AWS resource type is identified in this finding as the potential target of an adversary. Currently, GuardDuty can generate findings for the resource types that are listed in the GuardDuty active finding types.
- **ThreatFamilyName** - describes the overall threat or potential malicious activity that GuardDuty is detecting. For example, a value of **NetworkPortUnusual** indicates that an EC2 instance identified in the GuardDuty finding has no prior history of communications on a particular remote port that also is identified in the finding.
- **DetectionMechanism** - describes the method in which GuardDuty detected the finding. This can be used to indicate a variation on a common finding type or a finding that GuardDuty used a specific mechanism to detect. For example, **Backdoor:EC2/DenialOfService.Tcp** indicates denial of service (DoS) was detected over TCP. The UDP variant is **Backdoor:EC2/DenialOfService.Udp**.

  A value of **.Custom** indicates that GuardDuty detected the finding based on your custom threat lists. For more information, see Entity lists and IP address lists.

  A value of **.Reputation** indicates that GuardDuty detected the finding using a domain reputation score model. For more information, see How AWS tracks the cloud's biggest security threats and helps shut them down.
- **Artifact** - describes a specific resource that is owned by a tool that is used in the malicious activity. For example, **DNS** in the finding type CryptoCurrency:EC2/BitcoinTool.B!DNS indicates that an Amazon EC2 instance is communicating with a known Bitcoin-related domain.

> **Note**
>
> Artifact is optional and may not be available for all GuardDuty finding types.

Threat Purposes

In GuardDuty a *threat purpose* describes the primary purpose of a threat, an attack type, or a stage of a potential attack. For example, some threat purposes, such as **Backdoor**, indicate a type of attack. However some threat purposes, such as **Impact** align with MITRE ATT&CK tactics. The MITRE ATT&CK tactics indicate different phases in an adversary's attack cycle. In the current release of GuardDuty, ThreatPurpose can have the following values:

Backdoor

This value indicates that an adversary has compromised an AWS resource and altered the resource so that it is capable of contacting its home command and control (C&C) server to receive further instructions for malicious activity.

Behavior

This value indicates that GuardDuty has detected activity or activity patterns that are different from the established baseline for the AWS resources involved.

CredentialAccess

This value indicates that GuardDuty has detected activity patterns that an adversary may use to steal credentials, such as passwords, usernames, and access keys, from your environment. This threat purpose is based on MITRE ATT&CK tactics.

Cryptocurrency

This value indicates that GuardDuty has detected that an AWS resource in your environment is hosting software that is associated with cryptocurrencies (for example, Bitcoin).

DefenseEvasion

This value indicates that GuardDuty has detected activity or activity patterns that an adversary may use to avoid detection while infiltrating your environment. This threat purpose is based on MITRE ATT&CK tactics

Discovery

This value indicates that GuardDuty has detected activity or activity patterns that an adversary may use to expand their knowledge of your systems and internal networks. This threat purpose is based on MITRE ATT&CK tactics.

Execution

This value indicates that GuardDuty has detected that an adversary may try to run or has already run malicious code to explore the AWS environment, or steal data. This threat purpose is based on MITRE ATT&CK tactics .

Exfiltration

This value indicates that GuardDuty has detected activity or activity patterns that an adversary may use when attempting to steal data from your environment. This threat purpose is based on MITRE ATT&CK tactics.

Impact

This value indicates that GuardDuty has detected activity or activity patterns that suggest that an adversary is attempting to manipulate, interrupt, or destroy your systems and data. This threat purpose is based on MITRE ATT&CK tactics.

InitialAccess

This value is commonly associated with the initial access stage of an attack when an adversary is attempting to establish access to your environment. This threat purpose is based on MITRE ATT&CK tactics.

Pentest

Sometimes owners of AWS resources or their authorized representatives intentionally run tests against AWS applications to find vulnerabilities, such as open security groups or access keys that are overly-permissive. These pen tests are done in an attempt to identify and lock down vulnerable resources before they are discovered by adversaries. However, some of the tools used by authorized pen testers are freely available and therefore can be used by unauthorized users or adversaries to run probing tests. Although GuardDuty can't identify the true purpose behind such activity, the **Pentest** value indicates that GuardDuty is detecting such activity, that it is similar to the activity generated by known pen testing tools, and that it could indicate malicious probing of your network.

Persistence

This value indicates that GuardDuty has detected activity or activity patterns that an adversary may use to try and maintain access to your systems even if their initial access route is cut off. For example, this could include creating a new IAM user after gaining access through an existing user's compromised credentials. When the existing user's credentials are deleted, the adversary will retain access on the new user that was not detected as part of the original event. This threat purpose is based on MITRE ATT&CK tactics.

Policy

This value indicates that your AWS account is exhibiting behavior that goes against the recommended security best practices. For example, unintended modification of permission policies associated with your AWS resources or environment, and use of privileged accounts that should have little to no usage.

PrivilegeEscalation

This value informs you that the involved principal within your AWS environment is exhibiting behavior that an adversary may use to gain higher-level permissions to your network. This threat purpose is based on MITRE ATT&CK tactics.

Recon

This value indicates that GuardDuty has detected activity or activity patterns that an adversary may use when preforming reconnaissance of your environment to determine how they can broaden their access or utilize your resources. For example, this activity can include scoping out vulnerabilities in your AWS environment by probing ports, making API calls, listing users, and listing database tables among others.

Stealth

This value indicates that an adversary is actively trying to hide their actions. For example, they might use an anonymizing proxy server, making it extremely difficult to gauge the true nature of the activity.

Trojan

This value indicates that an attack is using Trojan programs that silently carry out malicious activity. Sometimes this software takes on an appearance of a legitimate program. Sometimes users accidentally run this software. Other times this software might run automatically by exploiting a vulnerability.

UnauthorizedAccess

This value indicates that GuardDuty is detecting suspicious activity or a suspicious activity pattern by an unauthorized individual.

GuardDuty malware detection scan engine

Amazon GuardDuty has an internally built and managed scan engine and a third-party vendor. Both use indicators of compromise (IoCs) sourced from various internal feeds that have visibility across different kinds of malware that may target AWS. GuardDuty also has detection definitions that are based on YARA rules added by our security engineers, and detections based on heuristic and machine learning (ML) models. When scanning Amazon S3 objects, GuardDuty Malware Protection produces consistent results when scanning the same object multiple times with the same scan definitions and engines. Signature-based detection not only includes matching of bytes but also a snippet of code that is potentially complex, and the scanner can parse content and make decisions.

The malware scan engine doesn't perform live behavioral analysis, where malware detonation monitors the sample as it executes in a real system. The GuardDuty solution is primarily a file-based detection. For detecting file-less malware, GuardDuty provides an agent-based solution, such as Runtime Monitoring for Amazon EKS, Amazon EC2, and Amazon ECS (including AWS Fargate).

With no restriction on the file formats that GuardDuty scans for malware, the scan engines that it uses can detect different types of malware, such as cryptominers, ransomware, and webshells. The fully managed GuardDuty scan engine continuously updates the list of malware signatures every 15 minutes.

The scan engine is a part of GuardDuty threat intelligence system that uses an internal malware detonation component. This generates new threat intelligence by independently collecting malware and benign samples from multiple sources. The file hash IoC type from the threat intelligence system further feeds into malware scan engine to detect malware based on known bad file hashes.

Generating sample findings in GuardDuty

Amazon GuardDuty helps you generate sample findings to visualize and understand the various finding types that it can generate. When you generate sample findings, GuardDuty populates your current findings list with one sample for each supported finding type, including attack sequence finding types.

The generated samples are approximations populated with placeholder values. These samples may look different from real findings for your environment, but you can use them to test various configurations for GuardDuty, such as your EventBridge events or filters. For a list of available values for finding types, see GuardDuty finding types table.

Generating sample findings through the GuardDuty console or API

Choose your preferred access method to generate sample findings.

> **Note**
>
> The GuardDuty console helps you generate one of each finding type. To generate one or more specific finding types, perform the associated API/CLI steps.

Console

Use the following procedure to generate sample findings. This process generates one sample finding for each GuardDuty finding type.

1. Open the GuardDuty console at https://console.aws.amazon.com/guardduty/.
2. In the navigation pane, choose **Settings**.
3. On the **Settings** page, under **Sample findings**, choose **Generate sample findings**.
4. In the navigation pane, choose **Findings**. The sample findings are displayed on the **Current findings** page with the prefix **[SAMPLE]**.

API/CLI

You can generate a single sample finding matching any of the GuardDuty finding types through the CreateSampleFindings API, the available values for finding types are listed in GuardDuty finding types table.

This is useful for the testing of CloudWatch Events rules or automation based on findings. The following example shows how to generate a single sample finding of the `Backdoor:EC2/DenialOfService.Tcp` type using the AWS CLI.

To find the `detectorId` for your account and current Region, see the **Settings** page in the https://console.aws.amazon.com/guardduty/ console, or run the ListDetectors API.

```
aws guardduty create-sample-findings --detector-id 12abc34d567e8fa901bc2d34e56789f0
 --finding-types Backdoor:EC2/DenialOfService.Tcp
```

The title of sample findings generated through these methods always begins with **[SAMPLE]** in the console. Sample findings have a value of `"sample": true` in the **additionalInfo** section of the finding JSON details.

To understand the finding details, such as finding severity and potentially compromised resource, associated with the generated findings, see Severity levels of GuardDuty findings and Finding details.

To generate some common findings based on a simulated activity in a dedicated and isolated AWS account within your environment, see Test GuardDuty findings in dedicated accounts.

Test GuardDuty findings in dedicated accounts

Use this document to run a tester script that generates GuardDuty findings against test resources that will be deployed in your AWS account. You can perform these steps when you want to understand and learn about certain GuardDuty finding types and how the finding details look for actual resources in your account. This experience is different from generating Sample findings. For more information about the experience of testing GuardDuty findings, see Considerations.

Contents

- Considerations
- GuardDuty findings tester script can generate
- Step 1 - Prerequisites
- Step 2 - Deploy AWS resources
- Step 3 - Run tester scripts
- Step 4 - Clean up AWS test resources
- Troubleshooting common issues

Considerations

Before you proceed, take the following considerations into account:

- GuardDuty recommends deploying the tester in a dedicated non-production AWS account. This approach will ensure that you are able to properly identify GuardDuty findings generated by the tester. Additionally, the GuardDuty tester deploys a variety of resources which may require IAM permissions beyond what is allowed in other accounts. Using a dedicated account ensures that permissions can be properly scoped with a clear account boundary.
- The tester script generates over 100 GuardDuty findings with different AWS resource combinations. Presently, this does't include all the GuardDuty finding types. For a list of finding

types that you can generate with this tester script, see GuardDuty findings tester script can generate.

Note

To visualize *Attack sequence finding types*, the tester script generates only AttackSequence:EKS/CompromisedCluster and AttackSequence:S3/CompromisedData. To visualize and understand AttackSequence:IAM/CompromisedCredentials, you can generate Sample findings in your account.

- For the GuardDuty tester to work as expected, GuardDuty needs to be enabled in the account where the tester resources are deployed. Depending on the tests that will run, the tester evaluates whether or not the appropriate GuardDuty protection plans are enabled. For any protection plan that is not enabled, GuardDuty will request permission to enable the necessary protection plans long enough for GuardDuty to perform the tests that will generate findings. Later, GuardDuty will disable the protection plan once the testing is complete.

  Enabling GuardDuty for the first time

  When GuardDuty gets enabled in your dedicated account for the first time in a specific Region, your account will be automatically enrolled in a 30-day free trial.

  GuardDuty offers optional protection plans. At the time of enabling GuardDuty, certain protection plans also get enabled and are included in the GuardDuty 30-day free trial. For more information, see Using GuardDuty 30-day free trial.

  GuardDuty is already enabled in your account prior to running the tester script

  When GuardDuty is already enabled, then based on the parameters, the tester script will check the configuration status of certain protection plans and other account level settings that are required to generate the findings.

  By running this tester script, certain protection plans may get enabled for the first time in your dedicated account in a Region. This will start the 30-day free trial for that protection plan. For information about free trial associated with each protection plan, see Using GuardDuty 30-day free trial.

- As long as the GuardDuty tester infrastructure is deployed, you may occasionally receive UnauthorizedAccess:EC2/TorClient findings from the PenTest instance.

GuardDuty findings tester script can generate

Presently, the tester script generates following finding types that are related to Amazon EC2, Amazon EKS, Amazon S3, IAM, and EKS audit logs:

- AttackSequence:EKS/CompromisedCluster
- AttackSequence:S3/CompromisedData
- Backdoor:EC2/C&CActivity.B!DNS
- Backdoor:EC2/DenialOfService.Dns
- Backdoor:EC2/DenialOfService.Udp
- CryptoCurrency:EC2/BitcoinTool.B!DNS
- Impact:EC2/AbusedDomainRequest.Reputation
- Impact:EC2/BitcoinDomainRequest.Reputation
- Impact:EC2/MaliciousDomainRequest.Reputation
- Impact:EC2/SuspiciousDomainRequest.Reputation
- Recon:EC2/Portscan
- Trojan:EC2/BlackholeTraffic!DNS
- Trojan:EC2/DGADomainRequest.C!DNS
- Trojan:EC2/DNSDataExfiltration
- Trojan:EC2/DriveBySourceTraffic!DNS
- Trojan:EC2/DropPoint!DNS
- Trojan:EC2/PhishingDomainRequest!DNS
- UnauthorizedAccess:EC2/MaliciousIPCaller.Custom
- UnauthorizedAccess:EC2/RDPBruteForce
- UnauthorizedAccess:EC2/SSHBruteForce
- PenTest:IAMUser/KaliLinux
- Recon:IAMUser/MaliciousIPCaller.Custom
- Recon:IAMUser/TorIPCaller
- Stealth:IAMUser/CloudTrailLoggingDisabled
- Stealth:IAMUser/PasswordPolicyChange

- UnauthorizedAccess:IAMUser/InstanceCredentialExfiltration.OutsideAWS
- UnauthorizedAccess:IAMUser/MaliciousIPCaller.Custom
- UnauthorizedAccess:IAMUser/TorIPCaller
- Discovery:Kubernetes/MaliciousIPCaller.Custom
- Discovery:Kubernetes/SuccessfulAnonymousAccess
- Discovery:Kubernetes/TorIPCaller
- Execution:Kubernetes/ExecInKubeSystemPod
- Impact:Kubernetes/MaliciousIPCaller.Custom
- Persistence:Kubernetes/ContainerWithSensitiveMount
- Policy:Kubernetes/AdminAccessToDefaultServiceAccount
- Policy:Kubernetes/AnonymousAccessGranted
- PrivilegeEscalation:Kubernetes/PrivilegedContainer
- UnauthorizedAccess:Lambda/MaliciousIPCaller.Custom
- Discovery:S3/MaliciousIPCaller.Custom
- Discovery:S3/TorIPCaller
- PenTest:S3/KaliLinux
- Policy:S3/AccountBlockPublicAccessDisabled
- Policy:S3/BucketAnonymousAccessGranted
- Policy:S3/BucketBlockPublicAccessDisabled
- Policy:S3/BucketPublicAccessGranted
- Stealth:S3/ServerAccessLoggingDisabled
- UnauthorizedAccess:S3/MaliciousIPCaller.Custom
- UnauthorizedAccess:S3/TorIPCaller
- Backdoor:Runtime/C&CActivity.B!DNS
- CryptoCurrency:Runtime/BitcoinTool.B!DNS
- DefenseEvasion:Runtime/ProcessInjection.Ptrace
- DefenseEvasion:Runtime/ProcessInjection.VirtualMemoryWrite
- Execution:Runtime/ReverseShell
- Impact:Runtime/AbusedDomainRequest.Reputation

- Impact:Runtime/BitcoinDomainRequest.Reputation
- Impact:Runtime/MaliciousDomainRequest.Reputation
- Impact:Runtime/SuspiciousDomainRequest.Reputation
- PrivilegeEscalation:Runtime/ContainerMountsHostDirectory
- PrivilegeEscalation:Runtime/DockerSocketAccessed
- Trojan:Runtime/BlackholeTraffic!DNS
- Trojan:Runtime/DGADomainRequest.C!DNS
- Trojan:Runtime/DriveBySourceTraffic!DNS
- Trojan:Runtime/DropPoint!DNS
- Trojan:Runtime/PhishingDomainRequest!DNS

Step 1 - Prerequisites

To prepare your test environment, you will need the following items:

- **Git** – Install git command line tool based on the operating system that you use.

  This is required to clone the `amazon-guardduty-tester` repository.
- **AWS Command Line Interface** – An open source tool that enables you to interact with AWS services by using commands in your command-line shell. For more information, see Get started with AWS CLI in the *AWS Command Line Interface User Guide*.
- **AWS Systems Manager** – To initiate Session Manager sessions with your managed nodes by using AWS CLI you must install the Session Manager plugin on your local machine. For more information, see Install Session Manager plugin for AWS CLI in the *AWS Systems Manager User Guide*.
- **Node Package Manager (NPM)** – Install NPM to install all the dependencies.
- **Docker** – You must have Docker installed. For installation instructions, see the Docker website.

  To verify that Docker has been installed, run the following command and confirm there is an output similar to the following output:

  ```
  $ docker --version
  Docker version 19.03.1
  ```
- Subscribe to Kali Linux image in the *AWS Marketplace*.

Step 2 - Deploy AWS resources

This section provides a list of key concepts and the steps to deploy certain AWS resources in your dedicated account.

Concepts

The following list provides key concepts related to the commands that help you deploy the resources:

- **AWS Cloud Development Kit (AWS CDK) –** CDK is an open-source software development framework for defining cloud infrastructure in code and provisioning it through AWS CloudFormation. CDK supports a couple of programming languages to define reusable cloud components known as constructs. You can compose these together into stacks and apps. Then, you can deploy your CDK applications to AWS CloudFormation to provision or update your resources. For more information, see What is the AWS CDK? in the *AWS Cloud Development Kit (AWS CDK) Developer Guide*.
- **Bootstrapping –** It is the process of preparing your AWS environment for usage with AWS CDK. Before you deploy a CDK stack into an AWS environment, the environment must first be bootstrapped. This process of provisioning specific AWS resources in your environment that are used by AWS CDK is part of the steps that you will perform in the next section - Steps to deploy AWS resources.

  For more information about how bootstrapping works, see Bootstrapping in the *AWS Cloud Development Kit (AWS CDK) Developer Guide*.

Steps to deploy AWS resources

Perform the following steps to start deploying the resources:

1. Set up your AWS CLI default account and Region unless the dedicated account Region variables are manually set in the `bin/cdk-gd-tester.ts` file. For more information, see Environments in the *AWS Cloud Development Kit (AWS CDK) Developer Guide*.
2. Run the following commands to deploy the resources:

```
git clone https://github.com/awslabs/amazon-guardduty-tester && cd amazon-guardduty-
tester
npm install
```

```
cdk bootstrap
cdk deploy
```

The last command (`cdk deploy`) creates a AWS CloudFormation stack on your behalf. The name of this stack is **GuardDutyTesterStack**.

As a part of this script, GuardDuty creates new resources to generate GuardDuty findings in your account. It also adds the following tag key:value pair to the Amazon EC2 instances:

`CreatedBy:GuardDuty Test Script`

The Amazon EC2 instances also include the EC2 instances that host EKS nodes and ECS clusters.

Instance types

GuardDuty is designed to use cost-effective instance types that provide the minimum performance necessary to successfully carry out tests. Because of vCPU requirements, the Amazon EKS node group requires `t3.medium`, and because of increased network capacity required for DenialOfService finding tests, the driver node requires `m6i.large`. For all other tests, GuardDuty uses `t3.micro` instance type. For more information about instance types, see Available sizes in the *Amazon EC2 Instances Types Guide*.

Step 3 - Run tester scripts

This is a two-step process where you first need to start a session with test driver and then, run scripts to generate GuardDuty findings with specific resource combinations.

Part A - Start session with test driver

1. After your resources are deployed, save the Region code to a variable in your current terminal session. Use the following command and replace *`us-east-1`* with the Region code where you deployed the resources:

   ```
   $ REGION=us-east-1
   ```

2. The tester script is available only through AWS Systems Manager (SSM). To start an interactive shell on the tester host instance, query the host **InstanceId**.
3. Use the following command to begin your session for the tester script:

```
aws ssm start-session
  --region $REGION
  --document-name AWS-StartInteractiveCommand
  --parameters command="cd /home/ssm-user/py_tester && bash -l"
  --target $(aws ec2 describe-instances
    --region $REGION
    --filters "Name=tag:Name,Values=Driver-GuardDutyTester"
    --query "Reservations[].Instances[?State.Name=='running'].InstanceId"
    --output text)
```

Part B - Generate findings

The tester script is a Python-based program that dynamically builds a bash script to generate findings based on your input. You have flexibility to generate findings based on one or more AWS resource types, GuardDuty protection plans, Threat Purposes (tactics), Foundational data sources, or the section called “GuardDuty findings tester script can generate”.

Use the following command examples as reference, and run one or more commands to generate findings that you want to explore:

```
python3 guardduty_tester.py
python3 guardduty_tester.py --all
python3 guardduty_tester.py --s3
python3 guardduty_tester.py --tactics discovery
python3 guardduty_tester.py --ec2 --eks --tactics backdoor policy execution
python3 guardduty_tester.py --eks --runtime only
python3 guardduty_tester.py --ec2 --runtime only --tactics impact
python3 guardduty_tester.py --log-source dns vpc-flowlogs
python3 guardduty_tester.py --finding 'CryptoCurrency:EC2/BitcoinTool.B!DNS'
```

For more information about valid parameters, you can run the following help command:

```
python3 guardduty_tester.py --help
```

Part C - Review generated findings

Choose a preferred method to view the generated findings in your account.

GuardDuty console

1. Sign in to the AWS Management Console and open the GuardDuty console at https://console.aws.amazon.com/guardduty/.
2. In the navigation pane, choose **Findings**.
3. From the findings table, select a finding for which you want to view the details. This will open up the finding details panel. For information, see Understanding and generating Amazon GuardDuty findings.
4. If you want to filter these findings, use the resource tag key and value. For example, to filter the findings generated for the Amazon EC2 instances, use `CreatedBy`:`GuardDuty Test Script` tag key:value pair for **Instance tag key** and **Instance tag key**.

API

- Run ListFindings to view the findings for a specific detector ID. You can specific parameters to filter findings.

  To find the `detectorId` for your account and current Region, see the **Settings** page in the https://console.aws.amazon.com/guardduty/ console, or run the ListDetectors API.

AWS CLI

- Run the following AWS CLI command to view the generated findings and replace *us-east-1* and *12abc34d567e8fa901bc2d34EXAMPLE* with suitable values:

```
aws guardduty list-findings --region us-east-1 --detector-
id 12abc34d567e8fa901bc2d34EXAMPLE
```

  To find the `detectorId` for your account and current Region, see the **Settings** page in the https://console.aws.amazon.com/guardduty/ console, or run the ListDetectors API.

  For more information about the parameters that you can use to filter findings, see list-findings in the *AWS CLI Command Reference*.

Step 4 - Clean up AWS test resources

The account-level settings and other configuration status updates made during Step 3 - Run tester scripts return to the original state when the tester script concludes.

After you have run the tester script, you can choose to clean up the AWS test resources. You can choose to do this by using one of the following methods:

- Run the following command:

```
cdk destroy
```

- Delete the AWS CloudFormation stack with the name **GuardDutyTesterStack**. For information about steps, see Deleting a stack on the AWS CloudFormation console.

Troubleshooting common issues

GuardDuty has identified common issues and recommends troubleshooting steps:

- `Cloud assembly schema version mismatch` – Update AWS CDK CLI to a version compatible with the required cloud assembly version, or to the latest available version. For more information, see AWS CDK CLI compatibility.
- `Docker permission denied` – Add the dedicated account user to the **docker** or **docker-users** so that the dedicated account can run the commands. For more information about steps, see Daemon socket option.
- `Your requested instance type is not supported in your requested Availability Zone` – Some Availability zones don't support particular instance types. To identify which availability zones support your preferred instance type and reattempt to deploy AWS resources, perform the following steps:
  1. Choose a preferred method to determine which Availability zones support your instance type:

     Console

     To identify Availability zones that support preferred instance type

     1. Sign in to the AWS Management Console and open the Amazon EC2 console at https://console.aws.amazon.com/ec2/.
     2. By using the AWS Region selector in the upper-right corner of the page, choose the Region where you want to launch the instance.

3. In the navigation pane, under **Instances**, choose **Instance Types**.
4. From the **Instance types** table, choose a preferred instance type.
5. Under **Networking**, view the Regions listed under **Availability zones**.

   Based on this information, you might need to choose a new Region where you can deploy the resources.

AWS CLI

Run the following command to view a list of Availability zones. Make sure to specify your preferred instance type and the Region (*us-east-1*).

```
aws ec2 describe-instance-type-offerings --location-type availability-zone  --
filters Name=instance-type,Values=Preferred instance type --region us-east-1 --
output table
```

For more information about this command, see describe-instance-type-offerings in the *AWS CLI Command Reference*.

When running this command, if you receive an error, make sure you are using the latest version of AWS CLI. For more information, see Troubleshooting in the *AWS Command Line Interface User Guide*.

2. Attempt deploying the AWS resources again and specify an Availability zone that supports your preferred instance type.

To re-attempt deploying AWS resources

1. Set up the default Region in the `bin/cdk-gd-tester.ts` file.
2. To specify the Availability zone, open the `amazon-guardduty-tester/lib/common/network/vpc.ts` file.
3. In this file, replace `maxAzs: 2,` with `availabilityZones: ['us-east-1a', 'us-east-1c'],` where you must specify the Availability zones for your instance type.
4. Continue with the remaining steps under Steps to deploy AWS resources.

Viewing generated findings in GuardDuty console

When GuardDuty detects an activity that matches the pattern of a security issue, GuardDuty generates a finding. This finding is associated with a resource type that may have been compromised during this activity. You can view the details associated with each finding that GuardDuty generates.

If you are using a GuardDuty administrator account, you can view the generated findings on behalf of the member accounts. However, a member account can view the findings generated in their own account. A member account can't view the findings generated for other member accounts.

Steps to view findings in GuardDuty console

1. Open the GuardDuty console at https://console.aws.amazon.com/guardduty/.
2. In the left navigation pane, choose **Findings**.

   GuardDuty displays the findings in a tabular format. By default, this table is sorted in decreasing order based on the **Last seen** column value, displaying the most recent findings at the top.

   Findings with a sword icon ( ) represent an attack sequence finding.
3. To view details associated with a finding, select its **Title**. This will open the finding details side panel. For an attack sequence finding, this side panel includes a *summarized version* of the attack sequence, and to expand this view, choose **View details**.

   For information about the fields listed in this side panel, see Finding details.
4. **(Optional) to download finding JSON**

   a. Select the finding, and then choose the **Actions** menu.
   b. On the **Actions** menu, choose **View and export JSON**.
   c. On the **Findings JSON** window, choose **Download**.

   > **Note**
   >
   > In some cases, GuardDuty becomes aware that certain findings are false positives after they have been generated. GuardDuty provides a **Confidence** field in the

finding's JSON, and sets its value to zero. This way GuardDuty lets you know that you can safely ignore such findings.
Findings without the **Confidence** field are not considered false positives.

Navigating Findings page

This section provides key information about various elements on the **Findings** page. This will help you analyze the generated findings for threat analysis and response.

The following list explains **Findings** page elements that will help you better understand the generated findings:

- **Threat type**:

  Threat type includes individual GuardDuty findings and attack sequence findings. By default, the page displays **All findings**.

  To filter the findings table view, on the **Threat type** menu, choose one of the options **– Attack sequence findings only** or **Individual findings only**.
- **Resource and Count columns**:

  The **Resource** column in the findings table shows the name of the potentially compromised AWS resource. For an attack sequence finding, this column shows the number of potentially compromised AWS resources. To view the resource names, select the *number* under the **Resource** column.

  The **Count** column indicates the number of times GuardDuty observes a specific finding. When GuardDuty detects that an activity that matches a previously identified security issue, it increments the count for that specific finding. For an attack sequence finding, this column value indicates the total number of signals and findings involved in the generation of the finding.
- **Sorting findings by table columns**:

  If there is an *arrow* next to a column header, then you can sort the findings table based on the column. Select the column header to sort the findings in either increasing or decreasing order of the value in that column.
- **Filtering findings**:

Based on specific property attributes, such as `Account ID` and `Resource type`, you can further filter the findings table. For information about types of filters you can use, see Filtering GuardDuty findings.

- **Status and Saved rules**:

  The **Status** menu includes two values – **Current** and **Archived**. The default view is **Current** findings in the table.

  When you no longer want GuardDuty to generate a finding that matches a specific criteria, you can suppress that finding. GuardDuty archives that finding. When GuardDuty detects this finding again, you will not be notified of this observation. To specifically view archived findings, on the **Status** menu, choose **Archived**.

  Saved rules is a feature that helps you automatically filter and take actions on findings that match a specified criteria. Actions may include archiving findings or suppressing them from future notifications.

  For more information, see Suppression rules.

Severity levels of GuardDuty findings

Each GuardDuty finding has an assigned severity level and value that reflects the potential risk the finding could have to your environment, as determined by our security engineers. The value of the severity can fall anywhere within the 1.0 to 10.0 range, with higher values indicating greater security risk. To help you determine a response to a potential security issue that is highlighted by a finding, GuardDuty breaks down this range into *Critical*, *High*, *Medium*, and *Low* severity levels.

A finding of a particular type may have a different severity depending on the context specific to the finding. To view a consolidated list of default severity levels for all GuardDuty finding types, see GuardDuty active finding types.

The following sections explain defined severity levels for the GuardDuty findings.

Topics

- Critical severity
- High severity
- Medium severity

- Low severity

Critical severity

Value range: 9.0 - 10.0

Description: A critical severity level indicates that an attack sequence may be in progress or had recently happened. One or more AWS resources, such as IAM user sign-in credentials and Amazon S3 bucket, are potentially being compromised or may have already been compromised.

Recommendation: GuardDuty recommends that you prioritize triaging and remediating all critical severity findings because these issues can be a part of a ransomware attack and can escalate at any time. View details about the involved resources and start addressing the security issues. For more information, see Remediating findings.

High severity

Value range: 7.0 - 8.9

Description: A High severity level indicates that the resource in question (an Amazon EC2 instance or a set of IAM user sign-in credentials) is compromised and is actively being used for unauthorized purposes.

Recommendation: GuardDuty recommends that you treat any high severity finding security issue as a priority and take immediate remediation steps to prevent further unauthorized use of your resources. For example, clean up your Amazon EC2 instance or terminate it, or rotate the IAM credentials. Follow the steps in Remediating findings to remediate the finding.

Medium severity

Value range: 4.0 - 6.9

Description: A medium severity level indicates suspicious activity that deviates from normally observed behavior and, depending on your use case, may be indicative of a resource compromise.

Recommendation: GuardDuty recommends investigating the potentially impacted resource at your earliest convenience. Remediation steps will vary by resource and finding family. An establish approach is for you to confirm that the activity is authorized and consistent with your use case. If you cannot identify the cause, or confirm the activity was authorized, you should consider the resource compromised. Follow the steps in Remediating findings to remediate the finding.

Here are some things to consider when reviewing a medium level finding:

- Check if an authorized user has installed new software that changed the behavior of a resource (for example, allowed higher than normal traffic, or enabled communication on a new port).
- Check if an authorized user changed the control plane settings, for example, modified a security group setting.
- Run an anti-virus scan on the implicated resource to detect unauthorized software.
- Verify the permissions that are attached to the implicated IAM role, user, group, or set of credentials. These might have to be changed or rotated.

Low severity

Value range: 1.0 - 3.9

Description: A low severity level indicates attempted suspicious activity that did not compromise your environment, for example, a port scan or a failed intrusion attempt.

Recommendation: There is no immediate recommended action, but it is worth taking a note of this information as it may indicate someone is looking for weak points in your environment.

Finding details

In the Amazon GuardDuty console, you can view finding details in the finding summary section. Finding details vary based on the finding type.

There are two primary details that determine what kind of information is available for any finding. The first is the resource type, which can be `Instance`, `AccessKey`, `S3Bucket`, `S3Object`, `Kubernetes cluster`, `ECS cluster`, `Container`, `RDSDBInstance`, `RDSLimitlessDB`, or `Lambda`. The second detail that determines finding information is **Resource Role**. Resource role can be `Target`, meaning the resource was the target of suspicious activity. For instance type findings, resource role can also be `Actor`, which means that your resource was the actor carrying out suspicious activity. This topic describes some of the commonly available details for findings. For the section called “Runtime Monitoring finding types” and Malware Protection for S3 finding type, the resource role is not populated.

Topics

- Finding overview

- Resource
- Attack sequence finding details
- RDS database (DB) user details
- Runtime Monitoring finding details
- EBS volumes scan details
- Malware Protection for EC2 finding details
- Malware Protection for S3 finding details
- Action
- Actor or Target
- Geolocation details
- Additional information
- Evidence
- Anomalous behavior

Finding overview

A finding's **Overview** section contains the most basic identifying features of the finding, including the following information:

- **Account ID** – The ID of the AWS account in which the activity took place that prompted GuardDuty to generate this finding.
- **Count** – The number of times GuardDuty has aggregated an activity matching this pattern to this finding ID.
- **Created at** – The time and date when this finding was first created. If this value differs from **Updated at**, it indicates that the activity has occurred multiple times and is an ongoing issue.

> **Note**
>
> Timestamps for findings in the GuardDuty console appear in your local time zone, while JSON exports and CLI outputs display timestamps in UTC.

- **Finding ID** – A unique identifier for this finding type and set of parameters. New occurrences of activity matching this pattern will be aggregated to the same ID.

- **Finding type –** A formatted string representing the type of activity that triggered the finding. For more information, see GuardDuty finding format.
- **Region –** The AWS Region in which the finding was generated. For more information about supported Regions, see Regions and endpoints
- **Resource ID –** The ID of the AWS resource against which the activity took place that prompted GuardDuty to generate this finding.
- **Scan ID –** Applicable to findings when GuardDuty Malware Protection for EC2 is enabled, this is an identifier of the malware scan that runs on the EBS volumes attached to the potentially compromised EC2 instance or container workload. For more information, see Malware Protection for EC2 finding details.
- **Severity –** A finding's assigned severity level of either Critical, High, Medium, or Low. For more information, see Findings severity levels.
- **Updated at –** The last time this finding was updated with new activity matching the pattern that prompted GuardDuty to generate this finding.

Resource

The **Resource affected** gives details about the AWS resource that was targeted by the initiating activity. The information available varies based on resource type and action type.

Resource role – The role of the AWS resource that initiated the finding. This value can be **TARGET** or **ACTOR**, and represents whether your resource was the target of the suspicious activity or the actor that performed the suspicious activity.

Resource type – The type of the affected resource. If multiple resources were involved, a finding can include multiple resources types. The resource types are **Instance**, **AccessKey**, **S3Bucket**, **S3Object**, **KubernetesCluster**, **ECSCluster**, **Container**, **RDSDBInstance**, **RDSLimitlessDB**, and **Lambda**. Depending on the resource type, different finding details are available. Select a resource option tab to learn about the details available for that resource.

Instance

Instance details:

> **Note**
>
> Some instance details may be missing if the instance has already been stopped or if the underlying API invocation originated from an EC2 instance in a different Region when making a cross-Region API call.

- **Instance ID** – The ID of the EC2 instance involved in the activity that prompted GuardDuty to generate the finding.
- **Instance Type** – The type of the EC2 instance involved in the finding.
- **Launch Time** – The time and date that the instance was launched.
- **Outpost ARN** – The Amazon Resource Name (ARN) of AWS Outposts. Only applicable to AWS Outposts instances. For more information, see What is AWS Outposts? in the *User Guide for Outposts racks*.
- **Security Group Name** – The name of the Security Group attached to the involved instance.
- **Security Group ID** – The ID of the Security Group attached to the involved instance.
- **Instance state** – The current state of the targeted instance.
- **Availability Zone** – The AWS Region Availability Zone in which the involved instance is located.
- **Image ID** – The ID of the Amazon Machine Image used to build the instance involved in the activity.
- **Image Description** – A description of the ID of the Amazon Machine Image used to build the instance involved in the activity.
- **Tags** – A list of tags attached to this resource, listed in the format of `key:value`.

AccessKey

Access Key details:

- **Access key ID** – The Access key ID of the user engaged in the activity that prompted GuardDuty to generate the finding.
- **Principal ID** – The principal ID of the user engaged in the activity that prompted GuardDuty to generate the finding.
- **User type** – The type of user engaged in the activity that prompted GuardDuty to generate the finding. For more information, see CloudTrail userIdentity element.

- **User name –** The name of the user engaged in the activity that prompted GuardDuty to generate the finding.

S3Bucket

Amazon S3 bucket details:

- **Name –** The name of the bucket involved in the finding.
- **ARN –** The ARN of the bucket involved in the finding.
- **Owner –** The canonical user ID of the user that owns the bucket involved in the finding. For more information on canonical user IDs see AWS account identifiers.
- **Type –** The type of bucket finding, can be either **Destination** or **Source**.
- **Default server side encryption –** The encryption details for the bucket.
- **Bucket Tags –** A list of tags attached to this resource, listed in the format of `key:value`.
- **Effective Permissions –** An evaluation of all effective permissions and policies on the bucket that indicates whether the involved bucket is publicly exposed. Values can be **Public** or **Not public**.

S3Object

- **S3 object details –** Includes the following information about the scanned S3 object:
  - **ARN –** Amazon Resource Name (ARN) of the scanned S3 object.
  - **Key –** The name assigned to the file when it was created in S3 bucket.
  - **Version Id –** When you have enabled bucket versioning, this field indicates the version Id associated with the latest version of the scanned S3 object. For more information, see Using versioning in S3 buckets in the *Amazon S3 User Guide*.
  - **eTag –** Represents the specific version of the scanned S3 object.
  - **Hash –** Hash of the threat detected in this finding.
- **S3 bucket details –** Includes the following information about the Amazon S3 bucket associated with the scanned S3 object:
  - **Name –** Indicates the name of the S3 bucket that contains the object.
  - **ARN –** Amazon Resource Name (ARN) of the S3 bucket.
- **Owner –** Canonical Id of the owner of the S3 bucket.

EKSCluster

Kubernetes cluster details:

- **Name** – The name of the Kubernetes cluster.
- **ARN** – The ARN that identifies the cluster.
- **Created At** – The time and date when this cluster was created.

> **Note**
>
> Timestamps for findings in the GuardDuty console appear in your local time zone, while JSON exports and CLI outputs display timestamps in UTC.

- **VPC ID** – The ID of the VPC that is associated to your cluster.
- **Status** – The current status of the cluster.
- **Tags** – The metadata that you apply to the cluster to help you to categorize and organize them. Each tag consists of a key and an optional value, listed in the format `key:value`. You get to define both key and value.

  Cluster tags do not propagate to any other resource associated with the cluster.

Kubernetes workload details:

- **Type** – The type of Kubernetes workload, such as pod, deployment, and job.
- **Name** – The name of the Kubernetes workload.
- **Uid** – The unique ID of the Kubernetes workload.
- **Created at** – The time and date when this workload was created.
- **Labels** – The key-value pairs attached to the Kubernetes workload.
- **Containers** – The details of the container running as a part of Kubernetes workload.
- **Namespace** – The workload belongs to this Kubernetes namespace.
- **Volumes** – The volumes used by the Kubernetes workload.
  - **Host path** – Represents a preexisting file or directory on the host machine that the volume maps to.
  - **Name** – The name of the volume.

- **pod security context –** Defines the privilege and acess control settings for all containers in a pod.
- **Host network –** Set to `true` if the pods are included in the Kubernetes workload.

Kubernetes user details:

- **Groups –** Kubernetes RBAC (role-access based control) groups of the user involved in the activity that generated the finding.
- **ID –** Unique ID of the Kubernetes user.
- **Username –** Name of the Kubernetes user involved in the activity that generated the finding.
- **Session name –** Entity that assumed the IAM role with Kubernetes RBAC permissions.

ECSCluster

ECS cluster details:

- **ARN –** The ARN that identifies the cluster.
- **Name –** The name of the cluster.
- **Status –** The current status of the cluster.
- **Active services count –** The number of services that are running on the cluster in an `ACTIVE` state. You can view these services with [ListServices]
- **Registered container instances count –** The number of container instances registered into the cluster. This includes container instances in both `ACTIVE` and `DRAINING` status.
- **Running tasks count –** The number of tasks in the cluster that are in the `RUNNING` state.
- **Tags –** The metadata that you apply to the cluster to help you to categorize and organize them. Each tag consists of a key and an optional value, listed in the format `key:value`. You get to define both key and value.
- **Containers –** The details about the container that's associated with the task:
  - **Container name –** The name of the container.
  - **Container image –** The image of the container.
- **Task details –** The details of a task in a cluster.
  - **ARN –** The Amazon Resource Name (ARN) of the task.
  - **Definition ARN –** The Amazon Resource Name (ARN) of the task definition that creates the task.

- **Version** – The version counter for the task.
- **Task created at** – The Unix timestamp when the task was created.
- **Task started at** – The Unix timestamp when the task started.
- **Task started by** – The tag specified when a task is started.

Container

Container details:

- **Container runtime** – The container runtime (such as `docker` or `containerd`) used to run the container.
- **ID** – The container instance ID or full ARN entries for the container instance.
- **Name** – The name of the container.
- **Image** – The image of the container instance.
- **Volume mounts** – List of container volume mounts. A container can mount a volume under its file system.
- **Security context** – The container security context defines privilege and access control settings for a container.
- **Process details** – Describes the details of the process that is associated to the finding.

RDSDBInstance

RDSDBInstance details:

> **Note**
>
> This resource is available in RDS Protection findings related to the database instance.

- **Database Instance ID** – The identifier associated to the database instance that was involved in the GuardDuty finding.
- **Engine** – The database engine name of the database instance involved in the finding. Possible values are Aurora MySQL-Compatible or Aurora PostgreSQL-Compatible.
- **Engine version** – The version of the database engine that was involved in the GuardDuty finding.

- **Database cluster ID –** The identifier of the database cluster that contains the database instance ID involved in the GuardDuty finding.
- **Database instance ARN –** The ARN that identifies the database instance involved in the GuardDuty finding.

RDSLimitlessDB

RDSLimitlessDB details:

This resource is available in RDS Protection findings related to the supported engine version of Limitless Database.

- **DB shard group identifier –** The name associated with the Limitless DB shard group.
- **DB shard group resource ID –** The resource identifier of the DB shard group within the Limitless DB.
- **DB shard group ARN –** The Amazon Resource Name (ARN) that identifies the DB shard group.
- **Engine –** The identifier of the Limitless DB involved in the finding.
- **Engine version –** The version of the Limitless DB engine.
- **DB cluster identifier –** The name of the database cluster that is part of the Limitless DB.

For information about user and authentication details of the potentially impacted database, see RDS database (DB) user details.

Lambda

Lambda function details

- **Function name –** The name of the Lambda function involved in the finding.
- **Function version –** The version of the Lambda function involved in the finding.
- **Function description –** A description of the Lambda function involved in the finding.
- **Function ARN –** The Amazon Resource Name (ARN) of the Lambda function involved in the finding.
- **Revision ID –** The revision ID of the Lambda function version.
- **Role –** The execution role of the Lambda function involved in the finding.
- **VPC configuration –** The Amazon VPC configuration, including the VPC ID, security group, and subnet IDs associated with your Lambda function.

  - **VPC ID –** The ID of the Amazon VPC that is associated with the Lambda function involved in the finding.
  - **Subnet IDs –** The ID of the subnets that are associated with your Lambda function.
  - **Security Group –** The security group attached to the involved Lambda function. This includes the security group name and group ID.
- **Tags –** A list of tags attached to this resource, listed in the format of `key`:`value` pair.

Attack sequence finding details

GuardDuty provides details for each finding it generates in your account. These details help you understand the reasons behind the finding. This section focuses on details associated with Attack sequence finding types. This includes insights such as potentially impacted resources, timeline of events, indicators, signals, and endpoints involved in the finding.

To view details associated with signals that are GuardDuty findings, see the associated sections on this page.

In the GuardDuty console, when you select an attack sequence finding, the details side panel is divided into the following tabs:

- **Overview –** Provides a compact view of the attack sequence details, including signals, MITRE tactics, and potentially impacted resources.
- **Signals –** Displays a timeline of events that are involved in an attack sequence.
- **Resources –** Provides information about the potentially impacted resources, or the resources that are potentially at risk.

The following list provides descriptions associated with the attack sequence finding details.

Signals

A signal could be an API activity or a finding that GuardDuty uses to detect an attack sequence finding. GuardDuty considers the weak signals that don't present themselves as clear threat, piece them together, and correlate with individually generated findings. For more context, the **Signals** tab provides a timeline of the signals, as observed by GuardDuty.

Each signal, that is a GuardDuty finding, has it's own severity level and value assigned to it. In the GuardDuty console, you can select each signal to view the associated details.

Actors

Provides details about the threat actors in an attack sequence. For more information, see Actor in *Amazon GuardDuty API Reference*.

Endpoints

Provides details about the network endpoints that were used in this attack sequence. For more information, see NetworkEndpoint in *Amazon GuardDuty API Reference*. For information about how GuardDuty determines location, see Geolocation details.

Indicators

Includes observed data that matches the pattern of a security issue. This data specifies as to why GuardDuty there is an indication of a potentially suspicious activity. For example, when the indicator name is `HIGH_RISK_API`, this indicates an action commonly used by threat actors, or a sensitive action that may cause potential impact to an AWS account, such as accessing credentials or modifying a resource.

The following table includes a list of potential indicators and their descriptions:

Indicator name	Description
`ATTACK_TACTIC`	The MITRE tactics, such as **Discovery** and **Impact**.
`ATTACK_TECHNIQUE`	The MITRE technique used by the threat actor in an attack sequence. Examples include gaining access to resources and using them in an unintended way, and exploiting vulnerabilities.
`CRYPTOMINING_DOMAIN`	Indicates a domain name associated with cryptocurrency mining pools or infrastructure. For example, DNS queries or connections to these domains from container or Kubernetes environments may indicate unauthorized cryptomining activity. Indicates that the Autonomous System Number (ASN) was identified as anomalous, based on the user's historical baseline. For more information, see Anomalous behavior.
`CRYPTOMINING_IP`	Indicates an IP address associated with cryptocurrency mining pools or infrastructure. For example, connections to these addresses from

Indicator name	Description
	container or Kubernetes environments may indicate unauthorized cryptomining activity. Indicates that the Autonomous System Number (ASN) was identifie d as anomalous, based on the user's historical baseline. For more information, see Anomalous behavior.
CRYPTOMIN ING_PROCESS	Indicates a process identified as cryptocurrency mining software running within container or Kubernetes environments. For example, these processes may consume excessive CPU resources. Indicates that the Autonomous System Number (ASN) was identifie d as anomalous, based on the user's historical baseline. For more information, see Anomalous behavior.
HIGH_RISK_API	The AWS API that includes the AWS service name and `eventName` indicates an action commonly used by threat actors, or is a sensitive action that may cause potential impact to an AWS account, such as credential access or resource modification.
MALICIOUS _DOMAIN	Indicates a domain name with suspected threat intelligence indicatin g malicious intent. For example, this includes command and control (C&C) servers, malware distribution sites, or phishing domains contacted from container or Kubernetes environments. Indicates that the Autonomous System Number (ASN) was identifie d as anomalous, based on the user's historical baseline. For more information, see Anomalous behavior.
MALICIOUS_IP	The IP address has confirmed threat intelligence indicating malicious intent.

Indicator name	Description
MALICIOUS_PROCESS	Indicates a process suspected to be malicious based on threat intelligence or behavioral analysis. For example, this includes known malware, backdoors, or unauthorized tools executing within container or Kubernetes environments with malicious intent. Indicates that the Autonomous System Number (ASN) was identified as anomalous, based on the user's historical baseline. For more information, see Anomalous behavior.
SUSPICIOUS_NETWORK	The network is associated with known low reputation scores, such as risky virtual private network (VPN) providers and proxy services.
SUSPICIOUS_PROCESS	Indicates that the Autonomous System Number (ASN) was identified as anomalous, based on the user's historical baseline. For more information, see Anomalous behavior.
SUSPICIOUS_USER_AGENT	The user agent is associated with potentially known suspicious or exploited applications, such as Amazon S3 clients and attack tools.
TOR_IP	The IP address is associated with a Tor exit node.
UNUSUAL_API_FOR_ACCOUNT	Indicates that the AWS API was invoked anomalously, based on the account's historical baseline. For more information, see Anomalous behavior.
UNUSUAL_ASN_FOR_ACCOUNT	Indicates that the Autonomous System Number (ASN) was identified as anomalous, based on the account's historical baseline. For more information, see Anomalous behavior.
UNUSUAL_ASN_FOR_USER	Indicates that the Autonomous System Number (ASN) was identified as anomalous, based on the user's historical baseline. For more information, see Anomalous behavior.

MITRE tactics

This field specifies the MITRE ATT&CK tactics that the threat actor attempts through an attack sequence. GuardDuty uses the MITRE ATT&ACK framework that adds context to the entire attack sequence. The colors that the GuardDuty console uses to specify the threat purposes that have been used by the threat actor, align with the colors that indicate the critical, high, medium, and low Findings severity levels.

Network indicators

Indicators include a combination of network indicator values that explain why a network is indicative of a suspicious behavior. This section is applicable only when the **Indicator** includes `SUSPICIOUS_NETWORK` or `MALICIOUS_IP`. The following example shows how network indicators might be associated with an indicator, where:

- *`AnyCompany`* is an Autonomous System (AS).
- `TUNNEL_VPN`, `IS_ANONYMOUS`, and `ALLOWS_FREE_ACCESS` are the network indicators.

```
...{
    "key": "SUSPICIOUS_NETWORK",
    "values": [{
        "AnyCompany": [
            "TUNNEL_VPN",
            "IS_ANONYMOUS",
            "ALLOWS_FREE_ACCESS"
        ]
    }]
}
...
```

The following table includes the network indicator values and their description. These tags are added based on the threat intelligence GuardDuty collects from sources such as Spur

Network indicator value	Description
TUNNEL_VPN	Network or IP address is associated with a VPN tunnel type. This refers to a specific protocol that helps establishing a secure, encrypted connection between two points over a public network.
TUNNEL_PROXY	Network or IP address is associated with a Proxy tunnel type. This refers to a specific protocol that helps establishing a connection through a proxy server.

Network indicator value	Description
TUNNEL_RDP	Network or IP address is associated with using a method of encapsulating remote desktop (RDP) traffic within another protocol to enhance security, bypass network restrictions, or enable remote access through firewalls.
IS_ANONYMOUS	Network or IP address is associated with a known anonymous or proxy services. This may indicate potential suspicious activities hiding behind anonymous networks.
KNOWN_THREAT_OPERATOR	Network or IP address is associated with a known risky tunnel provider. This indicates that suspicious activity has been detected from an IP address that is linked to a VPN, proxy, or other tunneling services frequently used for malicious purposes.
ALLOWS_FREE_ACCESS	Network or IP address is associated with a tunnel operator that allows access to it's service without requiring authentication or payment. It might also include trial accounts or limited usage experiences offered by various online services.
ALLOWS_CRYPTO	Network or IP address is associated with a tunnel provider (such as VPN or proxy service) that exclusively accepts cryptocurrency or other digital currencies as the method of payment.
ALLOWS_TORRENTS	Network or IP address is associated with services or platforms that allow torrent traffic. Such services are often associated with supporting and using torrent, and copyright circumvention activities.
RISK_CALLBACK_PROXY	Network or IP address is associated with devices known to route traffic for residential proxies, malware proxies, or other callback proxy-type networks. This **doesn't imply** all activity on the network is proxy-related, but rather that the network has the capability to route traffic on behalf of these proxy networks.

Network indicator value	Description
RISK_GEO_ MISMATCH	This indicator suggests that the datacenter or hosting location of a network differs from the expected location of the users and devices behind it. If this indicator value is not present, it doesn't mean that there is no mismatch. It might imply that there is insufficient data to confirm the discrepancy.
IS_SCANNER	Network or IP address is associated with conducting persistent login attempts against web forms.
RISK_WEB_ SCRAPING	Network of IP address is associated with automated web clients and other programmatic web activities.
CLIENT_BE HAVIOR_FI LE_SHARING	Network or IP address is associated with client behavior indicative of file sharing activities, such as peer-to-peer (P2P) networks, or file sharing protocols.
CATEGORY_ COMMERCIA L_VPN	Network or IP address is associated with a tunnel operator that is categorized as a traditional Commercial Virtual Private Network (VPN) service operating within datacenter space.
CATEGORY_ FREE_VPN	Network or IP address is associated with a tunnel operator that is categorized as a completely free VPN service.
CATEGORY_ RESIDENTI AL_PROXY	Network or IP address is associated with a tunnel operator that is categorized as an SDK, malware, or get-paid-to sourced proxy service.
OPERATOR_XXX	The name of the service provider that is operating this tunnel.

RDS database (DB) user details

> **Note**
>
> This section is applicable to findings when you enable the RDS Protection feature in GuardDuty. For more information, see GuardDuty RDS Protection.

The GuardDuty finding provides the following user and authentication details of the potentially compromised database:

- **User –** The user name used to make the anomalous login attempt.
- **Application –** The application name used to make the anomalous login attempt.
- **Database –** The name of the database instance involved in the anomalous login attempt.
- **SSL –** The version of the Secure Socket Layer (SSL) used for the network.
- **Auth method –** The authentication method used by the user involved in the finding.

For information about the potentially compromised resource, see Resource.

Runtime Monitoring finding details

> **Note**
>
> These details may be available only if GuardDuty generates one of the GuardDuty Runtime Monitoring finding types.

This section contains the runtime details such as process details and any required context. Process details describe information about the observed process, and runtime context describes any additional information about the potentially suspicious activity.

Process details

- **Name –** The name of the process.
- **Executable path –** The absolute path of the process executable file.
- **Executable SHA-256 –** The SHA256 hash of the process executable.

- **Namespace PID –** The process ID of the process in a secondary PID namespace other than the host level PID namespace. For processes inside a container, it is the process ID observed inside the container.
- **Present working directory –** The present working directory of the process.
- **Process ID –** The ID assigned to the process by operating system.
- **startTime –** The time when the process started. This is in UTC date string format (`2023-03-22T19:37:20.168Z`).
- **UUID –** The unique ID assigned to the process by GuardDuty.
- **Parent UUID –** The unique ID of the parent process. This ID is assigned to the parent process by GuardDuty.
- **User –** The user that executed the process.
- **User ID –** The ID of the user that executed the process.
- **Effective user ID –** The effective user ID of the process at the time of the event.
- **Lineage –** Information about the ancestors of the process.
  - **Process ID –** The ID assigned to the process by operating system.
  - **UUID –** The unique ID assigned to the process by GuardDuty.
  - **Executable path –** The absolute path of the process executable file.
  - **Effective user ID –** The effective user ID of the process at the time of the event.
  - **Parent UUID –** The unique ID of the parent process. This ID is assigned to the parent process by GuardDuty.
  - **Start Time –** The time when the process started.
  - **Namespace PID –** The process ID of the process in a secondary PID namespace other than the host level PID namespace. For processes inside a container, it is the process ID observed inside the container.
  - **User ID –** The user ID of the user that executed the process.
  - **Name –** Name of the process.

Runtime context

From the following fields, a generated finding may include only those fields that are relevant to the finding type.

- **Mount Source –** The path on the host that is mounted by the container.

- **Mount Target** – The path in the container that is mapped to the host directory.
- **Filesystem Type** – Represents the type of the mounted filesystem.
- **Flags** – Represents options that control the behavior of the event involved in this finding.
- **Modifying Process** – Information about the process that created or modified a binary, script, or a library, inside a container at runtime.
- **Modified At** – The timestamp at which the process created or modified a binary, script, or library inside a container at runtime. This field is in the UTC date string format (`2023-03-22T19:37:20.168Z`).
- **Library Path** – The path to the new library that was loaded.
- **LD Preload Value** – The value of the `LD_PRELOAD` environment variable.
- **Socket Path** – The path to the Docker socket that was accessed.
- **Runc Binary Path** – The path to the `runc` binary.
- **Release Agent Path** – The path to the `cgroup` release agent file.
- **Command Line Example** – The example of the command line involved in the potentially suspicious activity.
- **Tool Category** – Category that the tool belongs to. Some of the examples are Backdoor Tool, Pentest Tool, Network Scanner, and Network Sniffer.
- **Tool Name** – The name of the potentially suspicous tool.
- **Script Path** – The path to the executed script that generated the finding.
- **Threat File Path** – The suspicious path for which the threat intelligence details were found.
- **Service Name** – The name of the security service that has been disabled.

EBS volumes scan details

Note

This section is applicable to findings when you turn on the GuardDuty-initiated malware scan in Malware Protection for EC2.

The EBS volumes scan provides details about the EBS volume attached to the potentially compromised EC2 instance or container workload.

- **Scan ID –** The identifier of the malware scan.
- **Scan started at –** The date and time when the malware scan started.
- **Scan completed at –** The date and time when the malware scan completed.
- **Trigger Finding ID –** The finding ID of the GuardDuty finding that initiated this malware scan.
- **Sources –** The potential values are `Bitdefender` and `Amazon`.

  For more information about the scan engine used to detect malware, see GuardDuty malware detection scan engine.
- **Scan detections –** The complete view of details and results for each malware scan.
  - **Scanned item count –** The total number of scanned files. It provides details such as `totalGb`, `files`, and `volumes`.
  - **Threats detected item count –** The total number of malicious `files` detected during the scan.
  - **Highest severity threat details –** The details of the highest severity threat detected during the scan and the number of malicious files. It provides details such as `severity`, `threatName`, and `count`.
  - **Threats detected by Name –** The container element grouping threats of all severity levels. It provides details such as `itemCount`, `uniqueThreatNameCount`, `shortened`, and `threatNames`.

Malware Protection for EC2 finding details

> **Note**
>
> This section is applicable to findings when you turn on the GuardDuty-initiated malware scan in Malware Protection for EC2.

When the Malware Protection for EC2 scan detects malware, you can view the scan details by selecting the corresponding finding on the **Findings** page in the https://console.aws.amazon.com/guardduty/ console. The severity of your Malware Protection for EC2 finding depends on the severity of the GuardDuty finding.

The following information is available under the **Threats detected** section in the details panel.

- **Name –** The name of the threat, obtained by grouping the files by detection.

- **Severity –** The severity of the threat detected.
- **Hash –** The SHA-256 of the file.
- **File path –** The location of the malicious file in the EBS volume.
- **File name –** The name of the file in which the threat was detected.
- **Volume ARN –** The ARN of the scanned EBS volumes.

The following information is available under the **Malware scan details** section in the details panel.

- **Scan ID –** The scan ID of the malware scan.
- **Scan started at –** The date and time when the scan started.
- **Scan completed at –** The date and time when the scan completed.
- **Files scanned –** The total number of scanned files and directories.
- **Total GB scanned –** The amount of storage scanned during the process.
- **Trigger finding ID –** The finding ID of the GuardDuty finding that initiated this malware scan.
- The following information is available under the **Volume details** section in the details panel.
  - **Volume ARN –** The Amazon Resource Name (ARN) of the volume.
  - **SnapshotARN –** The ARN of the snapshot of the EBS volume.
  - **Status –** The scan status of the volume, such as `Running`, `Skipped`, and `Completed`.
  - **Encryption type –** The type of encryption used to encrypt the volume. For example, `CMCMK`.
  - **Device name –** The name of the device. For example, `/dev/xvda`.

Malware Protection for S3 finding details

The following malware scan details are available when you enable both GuardDuty and Malware Protection for S3 in your AWS account:

- **Threats –** A list of threats detected during the malware scan.

> **Multiple potential threats in archive files**
>
> If you have an archive file with potentially multiple threats in it, Malware Protection for S3 reports only the first detected threat. After this, the scan status is marked as complete. GuardDuty generates the associated finding type and also sends EventBridge events that it generates. For more information about monitoring the Amazon S3

object scans using the EventBridge events, see the sample notification schema for **THREATS_FOUND** in S3 object scan result.

- **Item path** – A list of nested item path and hash details of the scanned S3 object.
  - **Nested item path** – Item path of the scanned S3 object where the threat was detected.

    The value of this field is available only if the top-level object is an archive and if threat is detected inside an archive.
  - **Hash** – Hash of the threat detected in this finding.
- **Sources** – The potential values are `Bitdefender` and `Amazon`.

  For more information about the scan engine used to detect malware, see GuardDuty malware detection scan engine.

Action

A finding's **Action** gives details about the type of activity that triggered the finding. The information available varies based on action type.

Action type – The finding activity type. This value can be **NETWORK_CONNECTION**, **PORT_PROBE**, **DNS_REQUEST**, **AWS_API_CALL**, or **RDS_LOGIN_ATTEMPT**. The information available varies based on action type:

- **NETWORK_CONNECTION** – Indicates that network traffic was exchanged between the identified EC2 instance and the remote host. This action type has the following additional information:
  - **Connection direction** – The network connection direction observed in the activity that prompted GuardDuty to generate the finding. The values can be one of the following:
    - **INBOUND** – Indicates that a remote host initiated a connection to a local port on the identified EC2 instance in your account.
    - **OUTBOUND** – Indicates that the identified EC2 instance initiated a connection to a remote host.
    - **UNKNOWN** – Indicates that GuardDuty could not determine the direction of the connection.
  - **Protocol** – The network connection protocol observed in the activity that prompted GuardDuty to generate the finding.
  - **Local IP** – The original source IP address of the traffic that triggered the finding. This info can be used to distinguish between the IP address of an intermediate layer through which traffic

flows, and the original source IP address of the traffic that triggered the finding. For example the IP address of an EKS pod as opposed to the IP address of the instance on which the EKS pod is running.

  - **Blocked** – Indicates whether the targeted port is blocked.

- **PORT_PROBE** – Indicates that a remote host probed the identified EC2 instance on multiple open ports. This action type has the following additional information:
  - **Local IP** – The original source IP address of the traffic that triggered the finding. This info can be used to distinguish between the IP address of an intermediate layer through which traffic flows, and the original source IP address of the traffic that triggered the finding. For example the IP address of an EKS pod as opposed to the IP address of the instance on which the EKS pod is running.
  - **Blocked** – Indicates whether the targeted port is blocked.
- **DNS_REQUEST** – Indicates that the identified EC2 instance queried a domain name. This action type has the following additional information:
  - **Protocol** – The network connection protocol observed in the activity that prompted GuardDuty to generate the finding.
  - **Blocked** – Indicates whether the targeted port is blocked.
- **AWS_API_CALL** – Indicates that an AWS API was invoked. This action type has the following additional information:
  - **API** – The name of the API operation that was invoked and thus prompted GuardDuty to generate this finding.

    > **Note**
    >
    > These operations can also include non-API events captured by AWS CloudTrail. For more information, see Non-API events captured by CloudTrail.

  - **User Agent** – The user agent that made the API request. This value tells you whether the call was made from the AWS Management Console, an AWS service, the AWS SDKs, or the AWS CLI.
  - **ERROR CODE** – If the finding was triggered by a failed API call this displays the error code for that call.
  - **Service name** – The DNS name of the service that attempted to make the API call that triggered the finding.

- **RDS_LOGIN_ATTEMPT** – Indicates that a login attempt was made to the potentially compromised database from a remote IP address.
  - **IP address** – The remote IP address that was used to make the potentially suspicious login attempt.

Actor or Target

A finding has an **Actor** section if the **Resource role** was `TARGET`. This indicates that your resource was targeted by suspicious activity, and the **Actor** section contains details about the entity that targeted your resource.

A finding has a **Target** section if the **Resource role** was `ACTOR`. This indicates that your resource was involved in suspicious activity against a remote host, and this section contains information on the IP or domain that your resource targeted.

The information available in the **Actor** or **Target** section can include the following:

- **Affiliated** – Details about whether the AWS account of the remote API caller is related to your GuardDuty environment. If this value is `true`, the API caller is affiliated to your account in some manner; if `false`, the API caller is from outside your environment.
- **Remote Account ID** – The account ID that owns the outbound IP address that was used to access the resource at the final network.
- **IP address** – The IP address involved in the activity that prompted GuardDuty to generate the finding.
- **Location** – Location information for the IP address involved in the activity that prompted GuardDuty to generate the finding.
- **Organization** – ISP organization information of the IP address involved in the activity that prompted GuardDuty to generate the finding.
- **Port** – The port number involved in the activity that prompted GuardDuty to generate the finding.
- **Domain** – The domain involved in the activity that prompted GuardDuty to generate the finding.
- **Domain with suffix** – The second- and top-level domain involved in an activity that potentially prompted GuardDuty to generate the finding. For a list of top-level and second-level domains, see public suffix list.

Geolocation details

GuardDuty determines the location and network of requests by using MaxMind GeoIP databases. MaxMind reports very high accuracy of their data at country level, although accuracy varies according to factors such as country and type of IP address.

For more information about MaxMind, see [MaxMind IP Geolocation](). If you believe any of the GeoIP data incorrect, submit a correction request to MaxMind at [MaxMind Correct GeoIP2 Data]().

Additional information

All findings have an **Additional information** section that can include the following information:

- **Threat list name** – The name of the threat list that includes the IP address or the domain name involved in the activity that prompted GuardDuty to generate the finding.
- **Sample** – A true or false value that indicates whether this is a sample finding.
- **Archived** – A true or false value that indicates whether this is finding has been archived.
- **Unusual** – Activity details that were not observed historically. These can include an unusual (previously not observed) user, location, time, bucket, login behavior, or ASN Org.
- **Unusual protocol** – The network connection protocol involved in the activity that prompted GuardDuty to generate the finding.
- **Agent details** – Details about the security agent that is currently deployed on the EKS cluster in your AWS account. This is only applicable to EKS Runtime Monitoring finding types.
  - **Agent version** – The version of the GuardDuty security agent.
  - **Agent Id** – The unique identifier of the GuardDuty security agent.

Evidence

Findings based on threat intelligence have an **Evidence** section that includes the following information:

- **Threat intelligence details** – The name of the threat list on which the recognized `Threat name` appears.
- **Threat name** – The name of the malware family or other identifier that is associated with the threat.
- **Threat file SHA256** – SHA256 of the file that generated the finding.

Anomalous behavior

Findings types that end in **AnomalousBehavior** indicate that the finding was generated by the GuardDuty anomaly detection machine learning (ML) model. The ML model evaluates all API requests to your account and identifies anomalous events that are associated with tactics used by adversaries. The ML model tracks various factors of the API request, such as the user that made the request, the location the request was made from, and the specific API that was requested.

Details about which factors of the API request are unusual for the CloudTrail user identity that invoked the request can be found in the finding details. The identities are defined by the [CloudTrail userIdentity Element](), and the possible values are: `Root`, `IAMUser`, `AssumedRole`, `FederatedUser`, `AWSAccount`, or `AWSService`.

In addition to the details available for all GuardDuty findings that are associated with API activity, **AnomalousBehavior** findings have additional details that are outlined in the following section. These details can be viewed in the console and are also available in the finding's JSON.

- **Anomalous APIs** – A list of API requests that were invoked by the user identity in proximity to the primary API request associated with the finding. This pane further breaks down the details of the API event in the following ways.
  - The first API listed is the primary API, which is the API request associated with the highest-risk observed activity. This is the API that triggered the finding and correlates to the attack stage of the finding type. This is also the API that is detailed under the **Action** section in the console, and in the finding's JSON.
  - Any other APIs listed are additional anomalous APIs from the listed user identity observed in proximity to the primary API. If there is only one API on the list, the ML model did not identify any additional API requests from that user identity as anomalous.
  - The list of APIs is divided based on whether an API was **successfully called**, or if the API was unsuccessfully called, meaning an error response was received. The type of error response received is listed above each unsuccessfully called API. Possible error response types are: `access denied`, `access denied exception`, `auth failure`, `instance limit exceeded`, `invalid permission - duplicate`, `invalid permission - not found`, and `operation not permitted`.
  - APIs are categorized by their associated service.
  - For more context, choose **Historical APIs** to view the details about the top APIs, to a maximum of 20, usually seen for both the user identity and all users within the account. The APIs are

marked **Rare (less than once a month)**, **Infrequent (a few times a month)**, or **Frequent (daily to weekly)**, depending on how often they are used within your account.

- **Unusual Behavior (Account) –** This section gives additional details about the profiled behavior for your account.

  > **Profiled behavior**
  >
  > GuardDuty continually learns about the activities within your account based on delivered events. These activities and their observed frequency is known as profiled behavior.

  The information tracked in this panel includes:

  - **ASN Org –** The Autonomous System Number (ASN) org that the anomalous API call was made from.
  - **User Name –** The name of the user that made the anomalous API call.
  - **User Agent–** The user agent used to make the anomalous API call. The user agent is the method used to make the call such as `aws-cli` or `Botocore`.
  - **User Type –** The type of user that made the anomalous API call. Possible values are `AWS_SERVICE`, `ASSUMED_ROLE`, `IAM_USER`, or `ROLE`.
  - **Bucket –** The name of the S3 bucket that is being accessed.
- **Unusual Behavior (User Identity) –** This section gives additional details about the profiled behavior for the **User Identity** involved with the finding. When a behavior isn't identified as historical, this means the GuardDuty ML model hasn't previously seen this user identity making this API call in this way within the training period. The following additional details about the **User Identity** are available:
  - **ASN Org –** The ASN Org the anomalous API call was made from.
  - **User Agent–** The user agent used to make the anomalous API call. The user agent is the method used to make the call such as `aws-cli` or `Botocore`.
  - **Bucket –** The name of the S3 bucket that is being accessed.
- **Unusual Behavior (Bucket) –** This section gives additional details about the profiled behavior for the S3 bucket associated with the finding. When a behavior isn't identified as historical, this means the GuardDuty ML model hasn't previously seen API calls made to this bucket in this way within the training period. The information tracked in this section includes:
  - **ASN Org –** The ASN Org the anomalous API call was made from.

  - **User Name –** The name of the user that made the anomalous API call.
  - **User Agent–** The user agent used to make the anomalous API call. The user agent is the method used to make the call such as `aws-cli` or `Botocore`.
  - **User Type –** The type of user that made the anomalous API call. Possible values are `AWS_SERVICE`, `ASSUMED_ROLE`, `IAM_USER`, or `ROLE`.

> **Note**
>
> For more context on historical behaviors, choose **Historical behavior** in either **Unusual behavior (Account)**, **User ID**, or **Bucket** section to view details about the expected behavior in your account for each of the following categories: **Rare (less than once a month)**, **Infrequent (a few times a month)**, or **Frequent (daily to weekly)**, depending on how often they are used within your account.

- **Unusual Behavior (Database) –** This section provides additional details about the profiled behavior for the database instance associated with the finding. When a behavior isn't identified as historical, it means that the GuardDuty ML model hasn't previously seen a login attempt made to this database instance in this way within the training period. The information tracked for this section in the finding panel includes:
  - **User name –** The user name used to make the anomalous login attempt.
  - **ASN Org –** The ASN Org that the anomalous login attempt was made from.
  - **Application name –** The application name used to make the anomalous login attempt.
  - **Database name –** The name of the database instance involved in the anomalous login attempt.

  The **Historical behavior** section provides more context on the previously observed **User names**, **ASN Orgs**, **Application names**, and **Database names** for the associated database. Each unique value has an associated count representing the number of times this value was observed in a successful login event.

- **Unusual behavior (Account Kubernetes cluster, Kubernetes namespace, and Kubernetes username) –** This section provides additional details about the profiled behavior for the Kubernetes cluster and namespace associated with the finding. When a behavior isn't identified as historical, it means that the GuardDuty ML model hasn't previously observed this account, cluster, namespace, or username in this way. The information tracked for this section in the finding panel includes:

- **Username –** The user that called the Kubernetes API associated with the finding.
- **Impersonated Username –** The user being impersonated by `username`.
- **Namespace –** The Kubernetes namespace within the Amazon EKS cluster where the action occurred.
- **User Agent –** The user agent associated with the Kubernetes API call. The user agent is the method used to make the call such as `kubectl`.
- **API –** The Kubernetes API called by `username` within the Amazon EKS cluster.
- **ASN Information –** The ASN information, such as Organization and ISP, associated with the IP address of the user making this call.
- **Day of week –** The day of the week when the Kubernetes API call was made.
- **Permission –** The Kubernetes verb and resource being checked for access to indicate whether or not the `username` can use the Kubernetes API.
- **Service Account Name –** The service account associated with the Kubernetes workload that provides an identity to the workload.
- **Registry –** The container registry associated with the container image that is deployed in the Kubernetes workload.
- **Image –** The container image, without the associated tags and digest, that is deployed in the Kubernetes workload.
- **Image Prefix Config –** The image prefix with the container and workload security configuration enabled, such as `hostNetwork` or `privileged`, for the container using the image.
- **Subject Name –** The subjects, such as a `user`, `group`, or `serviceAccountName` that is bound to a reference role in a `RoleBinding` or `ClusterRoleBinding`.
- **Role Name –** The name of the role that is involved in creation or modification of roles or the `roleBinding` API.

S3 volume-based anomalies

This section details the contextual information for S3 volume-based anomalies. The volume-based finding (Exfiltration:S3/AnomalousBehavior) monitors for unusual numbers of S3 API calls made to the S3 buckets by users, indicating potential data exfiltration. The following S3 API calls are monitored for volume-based anomaly detection.

- `GetObject`

- `CopyObject.Read`
- `SelectObjectContent`

The following metrics would help to build a baseline of usual behavior when an IAM entity accesses an S3 bucket. To detect data exfiltration, volume-based anomaly detection finding evaluates all the activities against the usual behavioral baseline. Choose **Historical behavior** in the **Unusual behavior (User Identity)**, **Observed Volume (User Identity)**, and **Observed Volume (Bucket)** sections to view the following metrics, respectively.

- Number of `s3-api-name` API calls invoked by the IAM user or IAM role (depends on which one was issued) associated with the affected S3 bucket over the past 24 hours.
- Number of `s3-api-name` API calls invoked by the IAM user or IAM role (depends on which one was issued) associated with all S3 buckets over the past 24 hours.
- Number of `s3-api-name` API calls across all IAM user or IAM role (depends on which one was issued) associated with the affected S3 bucket over the past 24 hours.

RDS login activity-based anomalies

This section details the count of login attempts performed by the unusual actor and is grouped by the result of the login attempts. The RDS Protection finding types identify anomalous behavior by monitoring the login events for unusual patterns of `successfulLoginCount`, `failedLoginCount`, and `incompleteConnectionCount`.

- **successfulLoginCount –** This counter represents the sum of successful connections (correct combination of login attributes) made to the database instance by the unusual actor. Login attributes include user name, password, and database name.
- **failedLoginCount –** This counter represents the sum of failed (unsuccessful) login attempts made to establish a connection to the database instance. This indicates that one or more attributes of the login combination, such as user name, password, or database name were incorrect.
- **incompleteConnectionCount –** This counter represents the number of connection attempts that can't be classified as successful or failed. These connections are closed before the database provides a response. For example, port scanning where the database port is connected but no piece of information is sent to the database, or the connection was aborted before the login completed in a successful or failed attempt.

GuardDuty finding aggregation

GuardDuty updates the generated findings dynamically. If GuardDuty detects a new activity related to the same security issue, then instead of creating a new finding, GuardDuty will update the original finding with the latest details. This behavior allows you to identify any ongoing issues, without the need to look through multiple similar reports, and reduces the overall volume of findings for known security issues.

For example, for UnauthorizedAccess:EC2/SSHBruteForce finding, multiple access attempts against your instance will be aggregated to the same finding ID, increasing the **Count** number in the finding's details. This is because that finding represents a single security issue with the instance indicating that the SSH port on the instance is not properly secured against this type of activity. However, if GuardDuty detects SSH access activity targeting a new instance in your environment, it will create a new finding with a unique finding ID to alert you to the fact that there is a security issue associated with the new resource.

When a finding is aggregated, it is updated with information from the latest occurrence of that activity. This means that in the above example, if your instance is the target of a brute force attempt from a new actor, the finding details will be updated to reflect the remote IP of the most recent source and older information will be replaced. Complete information about individual activity attempts will still be available in your CloudTrail logs or VPC Flow Logs.

The criteria that alert GuardDuty to generate a new finding instead of aggregating an existing one is dependent on the finding type. The aggregation criteria for each finding type is determined by our security engineers to provide an overview of distinct security issues within your account.

When GuardDuty generates an attack sequence finding type in your account, the finding will be aggregated only when you GuardDuty identifies the similar signals in the same sequence in your account. Otherwise, GuardDuty will generate another attack sequence.

Managing Amazon GuardDuty findings

GuardDuty offers several important features to help you sort, store, and manage your findings. These features will help you tailor findings to your specific environment, reduce noise from low value findings, and help you focus on threats to your unique AWS environment. Review the topics on this page to understand how you can use these features to increase the value of security findings in your environment.

Topics:

Summary dashboard in Amazon GuardDuty

Learn about the components of the summary dashboard available in the GuardDuty console.

Filtering findings in GuardDuty

Learn how to filter GuardDuty findings based on the criteria you specify.

Suppression rules in GuardDuty

Learn how to automatically filter the findings GuardDuty alerts you to through suppression rules. Suppression rules automatically archive findings based on filters.

Customizing threat detection with entity lists and IP address lists

Customize the GuardDuty monitoring scope using IP Lists and Threat Lists based on publicly-routable IP addresses. Trusted IP lists prevent non-DNS findings from being generated from IP's you consider trusted, while Threat Intel Lists will cause GuardDuty to alert you of activity from user-defined IPs.

Exporting generated findings to Amazon S3

Export the generated findings to an Amazon S3 bucket so that you can maintain records past the 90-day findings retention period in GuardDuty. Use this historical data to track potential suspicious activities in your account and evaluate whether the recommended remediation steps were successful.

Processing GuardDuty findings with Amazon EventBridge

Set up automatic notifications for GuardDuty findings through Amazon EventBridge events. You can also automate other tasks through EventBridge to help you respond to findings.

[Understanding CloudWatch Logs and reasons for skipping resources during Malware Protection for EC2 scan]

Learn how you can audit the CloudWatch Logs for GuardDuty Malware Protection for EC2 and what are the reasons because of which your impacted Amazon EC2 instance or Amazon EBS volumes may have been skipped during the scanning process.

[Reporting false positives in Malware Protection for EC2]

Learn how you can report potential false positive threat detections in Malware Protection for S3.

[Reporting S3 object scan result as false positive in Malware Protection for S3]

Learn how you can report potential false positive threat detections in Malware Protection for S3.

Summary dashboard in Amazon GuardDuty

The GuardDuty **Summary** dashboard provides an aggregated view of the GuardDuty findings generated in your AWS account in the current AWS Region.

If you're using a GuardDuty administrator account, the dashboard provides aggregated statistics and data for your account and member accounts in your organization.

Viewing Summary dashboard

1. Open the GuardDuty console at https://console.aws.amazon.com/guardduty/.

   GuardDuty displays the **Summary** dashboard by default when you open the console.
2. On the **Summary** page, choose the desired AWS Region from the Region selector in the top-right corner of the console.
3. From the date range selector menu, choose the date range for which you want to view the summary. By default, the dashboard displays the data for the present day, **Today**.

   > **Note**
   >
   > If no findings were generated during the selected date range, the dashboard will not have any data to display. You can refresh the dashboard, or adjust the date range.

Topics

- Overview
- Findings
- Most common finding types
- Findings by severity
- Accounts with most findings
- Resources with findings
- Least occurring findings
- Protection plans coverage

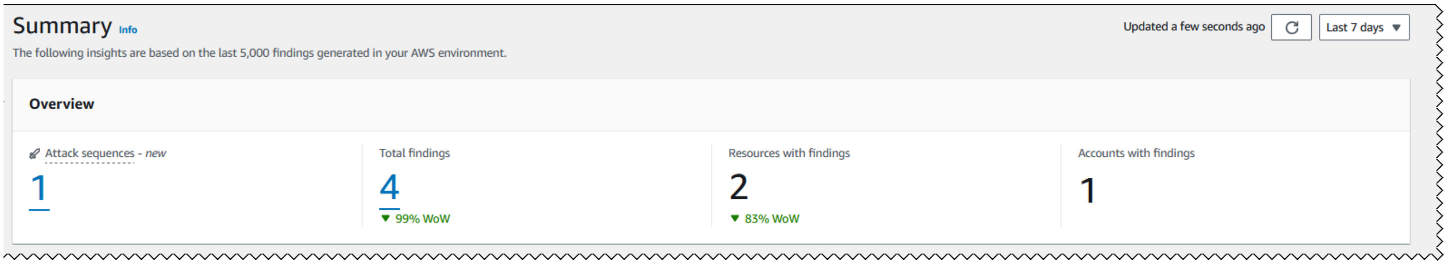
Overview

This section provides the following data:

- **Attack sequences**: Indicates the number of attack sequence findings that GuardDuty generated in your account in the current Region.

  GuardDuty detects potential multi-stage attacks in your account. You can select the *number* under **Attack sequences** to view its details on the **Findings** page.
- **Total findings**: Indicates the total number of findings generated in your account in the current Region. This includes both individual findings and attack sequence findings.
- **Resources with findings**: Indicates the number of resources that are associated to a finding, and have been potentially compromised.
- **Accounts with findings**: Indicates the number of accounts in which at least one finding was generated. If you're a standalone account, the value in this field is **1**.

For the time ranges **Last 7 days** and **Last 30 days**, the **Overview** pane may show the percentage difference in the findings generated week over week (WoW) or month over month (MoM), respectively. If no findings were generated in the week or the month before, then with no data to compare, the percentage difference may not be available.



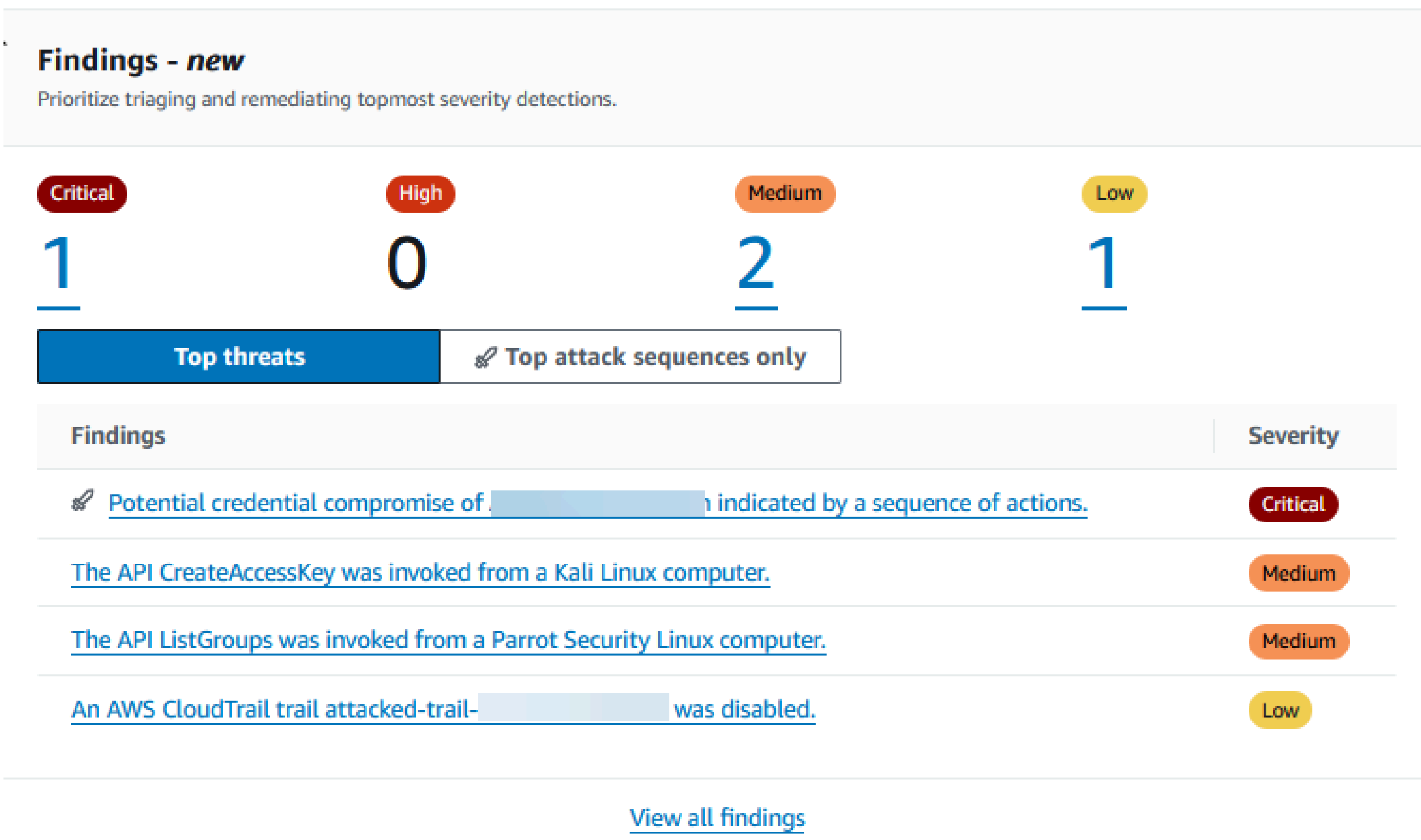
If you're a GuardDuty administrator account, all of these fields provide the summarized data across all the member accounts in your organization.

Findings

The **Findings** widget displays up to eight top findings. These findings are listed on the basis of their severity level, with *Critical* findings displayed first.

By default, you can view all the findings. To view only attack sequence findings data, turn on **Top attack sequences only**.

In this list, you can select any finding to view its details.



Most common finding types

This section provides a pie chart illustrating the top five most common finding types generated in the current Region. When hovering over each sector of the pie chart, you can observe the following:

- **Findings count**: Indicates the number of times this finding has been generated in the chosen date range.
- **Severity**: Indicates the severity level of the finding.
- **Percentage**: Indicates proportion of this finding type relative to the total.
- **Last generated**: Indicates how much time has passed since this finding type was last detected.

Findings by severity

This section displays a bar chart showing the total number of findings over the selected date range. The chart breaks down findings by severity (*Critical*, *High*, *Medium*, and *Low*), and helps you view the number of findings for specific dates within the range.

To view the counts for each severity level on a specific date, hover over the corresponding bar in the chart.

Accounts with most findings

This section provides the following data:

- **Account**: Indicates the AWS account ID where the finding was generated.
- **Finding count**: Indicates the number of times a finding was generated for this account ID.
- **Last generated**: Indicates how much time has passed since a finding type was last generated for this account ID.
- **Severity filter**: By default, the data is shown for the high severity finding types. Possible options for this field are **All severity**, **Critical severity**, **High severity**, and **Medium severity**.

Resources with findings

This section provides the following data:

- **Resource**: Shows the potentially impacted resource type and if this resource belongs to your account, you can access the quick link to view the resource details. If you're a GuardDuty administrator account, you can view the details of the potentially impacted resource by accessing the GuardDuty console with the credentials of the owner member account.
- **Account**: Indicates the AWS account ID to which this resource belongs.
- **Finding count**: Indicates the number of times that this resource was associated to a finding.
- **Last generated**: Indicates how much time has passed since a finding type associated to this resource was last generated.
- **Resource type filter**: By default, the data is shown for all the resource types. By using this filter, you can choose to view the data for a specific resource type, such as **Instance**, **AccessKey**, **Lambda**, and others.
- **Severity filter**: By default, the data is shown for **All severity**. By using this filter, you can choose to view the data for other severity levels. Possible options are **Critical severity**, **High severity**, **Medium severity**, and **All severity**.

Least occurring findings

This section highlights finding types that occur infrequently in your AWS environment. This widget is designed to help you identify and investigate potential emergent threat patterns.

This widget displays the following data:

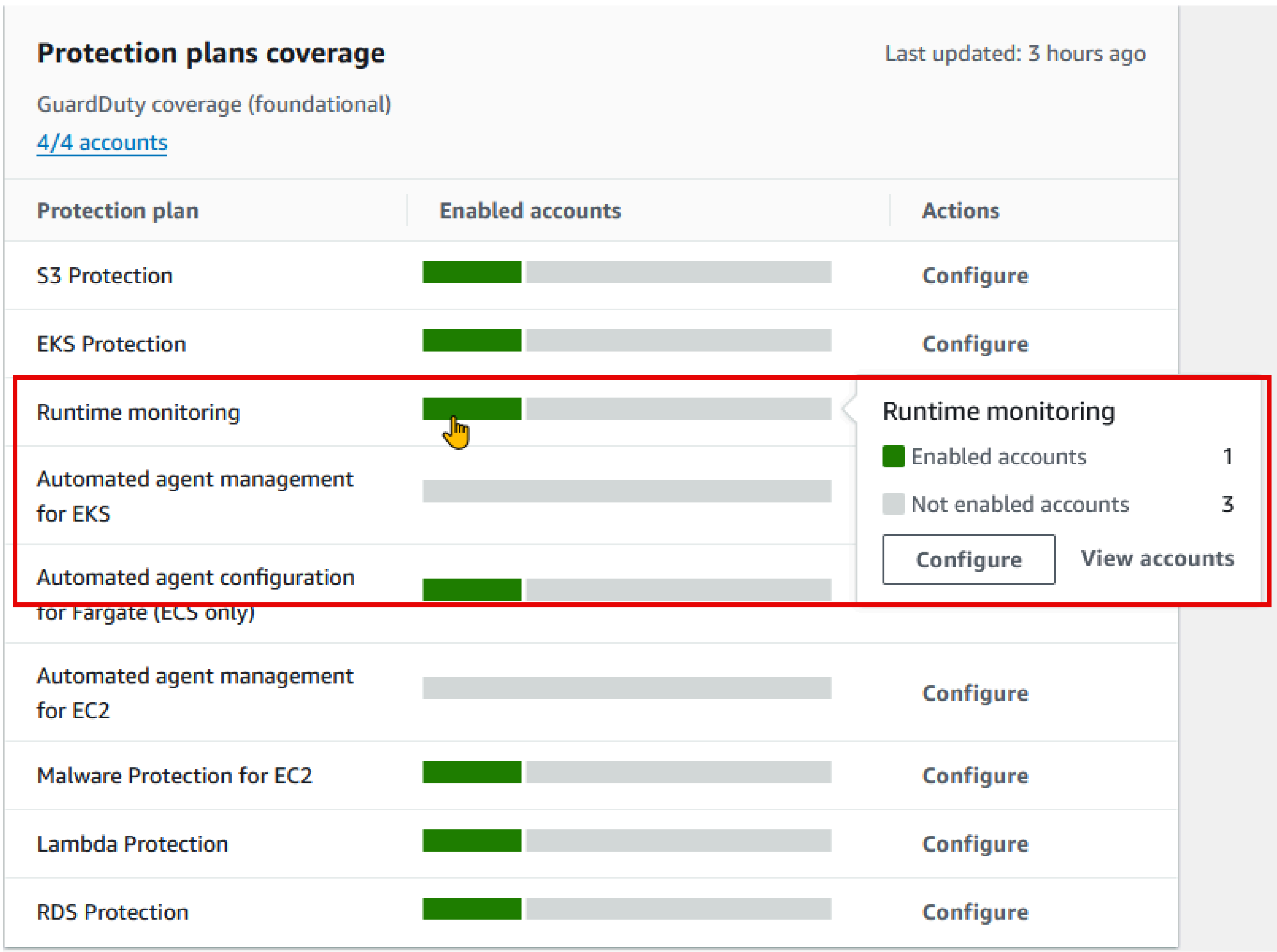
- **Finding type**: Shows the finding type name.
- **Finding count**: Indicates the number of times that this finding type was generated in the chosen time range.
- **Last generated**: Indicates how much time has passed since this finding type was last generated.
- **Severity filter**: By default, the data is shown for the high severity finding types. Possible options for this field are **Critical severity**, **High severity**, **Medium severity**, and **All severity**.

Protection plans coverage

This section displays statistics for the member accounts in your organization. It shows the number of member accounts that have enabled GuardDuty (foundational threat detection) in the current Region. Only a delegated GuardDuty administrator can view the statistics for the member accounts within their organization. When you create a new AWS organization, it might take up to 24 hours to generate the statistics for the entire organization.

How to use this widget

- **Configuration**: If a protection plan is not configured, choose **Configure** under the **Actions** column.
- **Viewing enabled accounts**: Hover over the bar in the **Enabled accounts** column to view how many accounts have enabled each protection plan. To further view account details, select the green bar, and choose **View accounts**.



Filtering findings in GuardDuty

A finding filter allows you to view findings that match the criteria you specify and filter out any unmatched findings. You can easily create finding filters using the Amazon GuardDuty console, or you can create them with the CreateFilter API using JSON. Review the following sections to understand how to create a filter in the console. To use these filters to automatically archive incoming findings, see Suppression rules in GuardDuty.

When you create filters, take the following list into consideration:

- GuardDuty **doesn't support** wild cards for filter criteria.
- You can specify a minimum of one attribute and up to a maximum of 50 attributes as the criteria for a particular filter.
- When you use the **Equals** or **Does not equals** operator to filter on an attribute value, such as Account ID, you can specify a maximum of 50 values.
- Each filter criteria attribute is evaluated as an `AND` operator. Multiple values for the same attribute are evaluated as `AND/OR`.
- For information about the maximum number of saved filters that you can create in an AWS account in each AWS Region, see GuardDuty quotas.

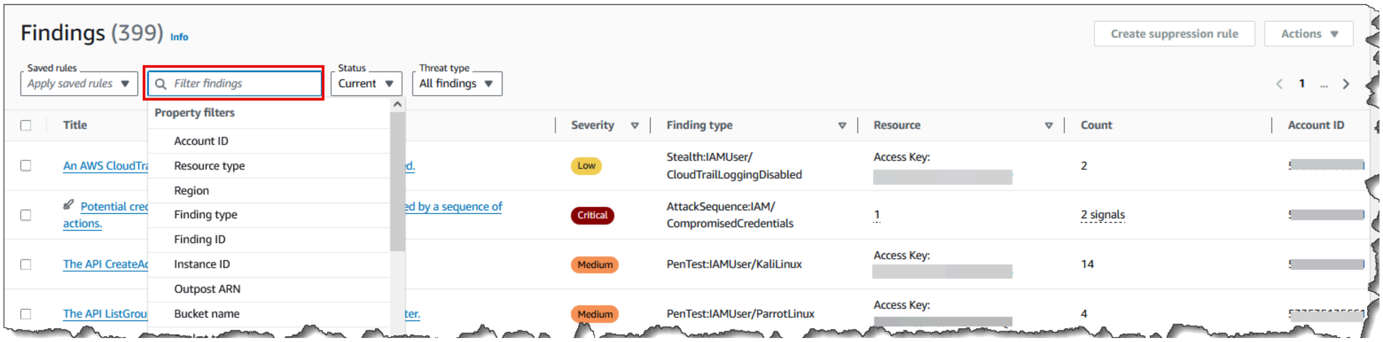
The following sections provide instructions on how to create and save filters using GuardDuty console, and API and CLI commands. Choose your preferred access method to proceed.

Creating and saving filter set in the GuardDuty console

Finding filters can be created and tested through the GuardDuty console. You can save filters created through the console for use in suppression rules or future filter operations. A filter is made up of at least one filter criteria, which consists of one filter attribute paired with at least one value.

To create and save filter criteria (console)

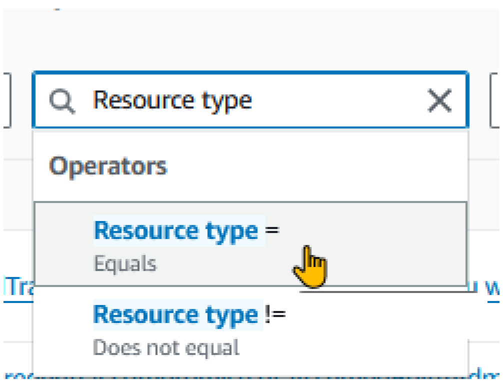
1. Sign in to the AWS Management Console and open the GuardDuty console at https://console.aws.amazon.com/guardduty/.
2. In the left navigation pane, choose **Findings**.
3. On the **Findings** page, select the *Filter findings* bar next to **Saved rules** menu. This will display an expanded list of **Property filters**.



4. From the expanded list of filters, select an attribute based on which you want to filter the findings table.

   For example, to view findings for which the potentially impacted resource is an **S3Bucket**, choose **Resource type**.

5. For **Operators**, choose one that will help you filter the findings to get the desired result. To continue the example from the previous step, choose **Resource type =**. This will display a list of resource types in GuardDuty.



   If your use case requires excluding specific findings, you can choose **Does not equal** or **!=** operator.

6. Specify the value for the selected property filter. If needed, choose **Apply**. To continue the example from the previous step, you can choose **S3Bucket**.

   This will display the findings that match with the applied filters.

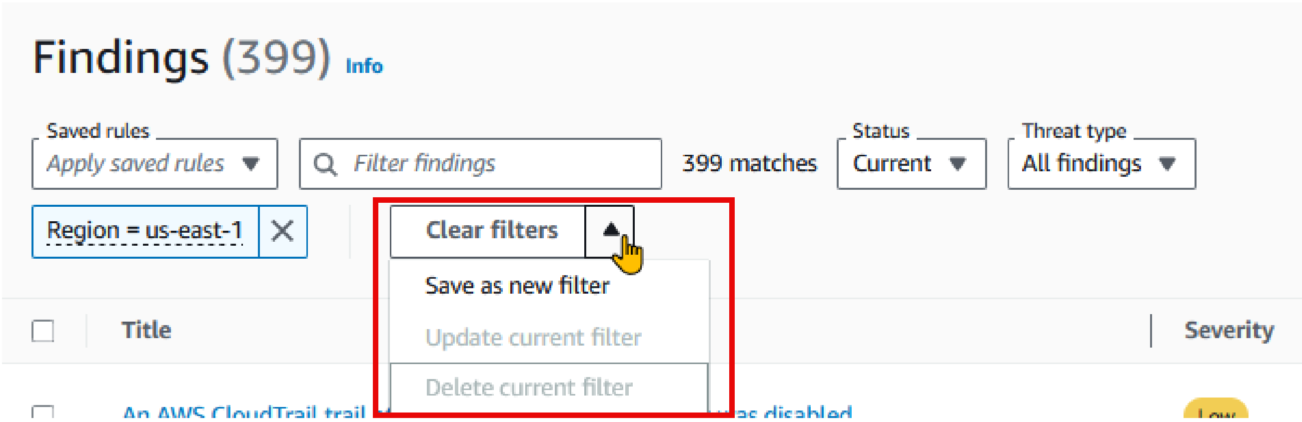
7. To add more than one filter criteria, repeat steps 3-6.

   For a complete list of attributes, see [Property filters in GuardDuty](#).

8. **(Optional) save the specified attributes and values as filters**

   To apply this filter combination again in the future, you can save the specified attributes and their values as a filter set.

   a. After you have created a filter criteria with one or more property filters, select the *arrow* in the **Clear filters** menu.



   b. Enter the filter set **Name**. The name must be 3-64 characters. Valid characters are a-z, A-Z, 0-9, period (.), hyphen (-), and underscore (_).

   c. The **Description** is optional. If you enter a description, it can have up to 512 characters.

   d. Choose **Create**.

Creating and saving filter set by using GuardDuty API and CLI

You can create and test the finding filters by using either API or CLI commands. A filter is made up of at least one filter criteria, which consists of one filter attribute paired with at least one value. You can save filters to create Suppression rules or to perform other filter operations later.

To create finding filters using API/CLI

- Run CreateFilter API by using the regional detector ID of the AWS account where you want to create a filter.

  To find the `detectorId` for your account and current Region, see the **Settings** page in the https://console.aws.amazon.com/guardduty/ console, or run the ListDetectors API.

- Alternatively, you can use the create-filter CLI to create and save the filter. You can use one or more filter criteria from Property filters in GuardDuty.

Use the following examples by replacing the placeholder values shown in red.

Example 1: Create a new filter to view all the findings that match a specific finding type

The following example creates a filter that matches all `PortScan` findings for an instance created from a specific image. The placeholder values are shown in red. Replace these values with suitable values for your account. For example, replace *12abc34d567e8fa901bc2d34EXAMPLE* with your regional detector ID.

```
aws guardduty create-filter \
--detector-id 12abc34d567e8fa901bc2d34EXAMPLE \
--name FilterExampleName \
--finding-criteria '{"Criterion": {"type": {"Equals": ["Recon:EC2/Portscan"]},
 "resource.instanceDetails.imageId": {"Equals":["ami-0a7a207083example"]}} }'
```

Example 2: Create a new filter to view all the findings that match severity levels

The following example creates a filter that matches all findings associated with the `HIGH` severity levels. The placeholder values are shown in red. Replace these values with suitable values for your account. For example, replace *12abc34d567e8fa901bc2d34EXAMPLE* with your regional detector ID.

```
aws guardduty create-filter \
--detector-id 12abc34d567e8fa901bc2d34EXAMPLE \
--name FilterExampleName \
--finding-criteria '{"Criterion": {"severity": {"Equals": ["7", "8"]}} }'
```

- For API/CLI, the Findings severity levels are represented as numerals. To filter the findings based on the severity levels, use the following values:
  - For `LOW` severity levels, use `{ "severity": { "Equals": ["1", "2", "3"] } }`
  - For `MEDIUM` severity levels, use `{ "severity": { "Equals": ["4", "5", "6"] } }`
  - For `HIGH` severity levels, use `{ "severity": { "Equals": ["7", "8"] } }`
  - For `CRITICAL` severity levels, use `{ "severity": { "Equals": ["9", "10"] } }`
  - For findings with multiple severity levels, use placeholder values similar to the following example: `{ "severity": { "Equals": ["7", "8", "9", "10"] } }`

    This example will show the findings that have either `HIGH` or `CRITICAL` severity levels.

> **Note**
>
> If you specify an example with only one numeric value instead of all the numeric values associated with a severity level, the API and CLI might show the filtered findings. When you use this saved filter set in the GuardDuty console, it will not work as expected. This is because the GuardDuty console considers the filter values as `CRITICAL`, `HIGH`, `MEDIUM`, and `LOW`. For example, a filter created with a CLI command that includes `{ "severity": { "Equals": ["9"] } }` is expected to show an appropriate output in API/CLI. However, this saved filter includes partial severity level when used in the GuardDuty console and will not show an expected output. This makes it necessary for the API and CLI to specify all the values associated with each severity level.

Property filters in GuardDuty

When you create filters or sort findings using the API operations, you must specify filter criteria in JSON. These filter criteria correlate to a finding's details JSON. The following table contains a list of the console display names for filter attributes and their equivalent JSON field names.

Console field name	JSON field name
Account ID	accountId
Finding ID	id
Region	region
Severity	severity You can filter the finding types based on the severity level of the finding types. For more information about severity values, see Severity levels of GuardDuty findings. If you use `severity` with API, AWS CLI, or AWS CloudFormation, it is assigned a numeric

Console field name	JSON field name
	value. For more information, see [findingCriteria]() in the *Amazon GuardDuty API Reference*.
Finding type	type
Updated at	updatedAt
Access Key ID	resource.accessKeyDetails.accessKeyId
Principal ID	resource.accessKeyDetails.principalId
Username	resource.accessKeyDetails.userName
User type	resource.accessKeyDetails.userType
IAM instance profile ID	resource.instanceDetails.iamInstanceProfile.id
Instance ID	resource.instanceDetails.instanceId
Instance image ID	resource.instanceDetails.imageId
Instance tag key	resource.instanceDetails.tags.key
Instance tag value	resource.instanceDetails.tags.value
IPv6 address	resource.instanceDetails.networkInterfaces.ipv6Addresses
Private IPv4 address	resource.instanceDetails.networkInterfaces.privateIpAddresses.privateIpAddress
Public DNS name	resource.instanceDetails.networkInterfaces.publicDnsName
Public IP	resource.instanceDetails.networkInterfaces.publicIp
Security group ID	resource.instanceDetails.networkInterfaces.securityGroups.groupId

Console field name	JSON field name
Security group name	resource.instanceDetails.networkInterfaces.securityGroups.groupName
Subnet ID	resource.instanceDetails.networkInterfaces.subnetId
VPC ID	resource.instanceDetails.networkInterfaces.vpcId
Outpost ARN	resource.instanceDetails.outpostARN
Resource type	resource.resourceType
Bucket permissions	resource.s3BucketDetails.publicAccess.effectivePermission
Bucket name	resource.s3BucketDetails.name
Bucket tag key	resource.s3BucketDetails.tags.key
Bucket tag value	resource.s3BucketDetails.tags.value
Bucket type	resource.s3BucketDetails.type
Action type	service.action.actionType
API called	service.action.awsApiCallAction.api
API caller type	service.action.awsApiCallAction.callerType
API Error Code	service.action.awsApiCallAction.errorCode
API caller city	service.action.awsApiCallAction.remoteIpDetails.city.cityName
API caller country	service.action.awsApiCallAction.remoteIpDetails.country.countryName

Console field name	JSON field name
API caller IPv4 address	service.action.awsApiCallAction.remoteIpDetails.ipAddressV4
API caller IPv6 address	service.action.awsApiCallAction.remoteIpDetails.ipAddressV6
API caller ASN ID	service.action.awsApiCallAction.remoteIpDetails.organization.asn
API caller ASN name	service.action.awsApiCallAction.remoteIpDetails.organization.asnOrg
API caller service name	service.action.awsApiCallAction.serviceName
DNS request domain	service.action.dnsRequestAction.domain
DNS request domain suffix	service.action.dnsRequestAction.domainWithSuffix
Network connection blocked	service.action.networkConnectionAction.blocked
Network connection direction	service.action.networkConnectionAction.connectionDirection
Network connection local port	service.action.networkConnectionAction.localPortDetails.port
Network connection protocol	service.action.networkConnectionAction.protocol
Network connection city	service.action.networkConnectionAction.remoteIpDetails.city.cityName
Network connection country	service.action.networkConnectionAction.remoteIpDetails.country.countryName

Console field name	JSON field name
Network connection remote IPv4 address	service.action.networkConnectionAction.remoteIpDetails.ipAddressV4
Network connection remote IPv6 address	service.action.networkConnectionAction.remoteIpDetails.ipAddressV6
Network connection remote IP ASN ID	service.action.networkConnectionAction.remoteIpDetails.organization.asn
Network connection remote IP ASN name	service.action.networkConnectionAction.remoteIpDetails.organization.asnOrg
Network connection remote port	service.action.networkConnectionAction.remotePortDetails.port
Remote account affiliated	service.action.awsApiCallAction.remoteAccountDetails.affiliated
Kubernetes API caller IPv4 address	service.action.kubernetesApiCallAction.remoteIpDetails.ipAddressV4
Kubernetes API caller IPv6 address	service.action.kubernetesApiCallAction.remoteIpDetails.ipAddressV6
Kubernetes namespace	service.action.kubernetesApiCallAction.namespace
Kubernetes API caller ASN ID	service.action.kubernetesApiCallAction.remoteIpDetails.organization.asn
Kubernetes API call request URI	service.action.kubernetesApiCallAction.requestUri
Kubernetes API status code	service.action.kubernetesApiCallAction.statusCode
Network connection local IPv4 address	service.action.networkConnectionAction.localIpDetails.ipAddressV4

Console field name	JSON field name
Network connection local IPv6 address	service.action.networkConnectionAction.localIpDetails.ipAddressV6
Protocol	service.action.networkConnectionAction.protocol
API call service name	service.action.awsApiCallAction.serviceName
API caller account ID	service.action.awsApiCallAction.remoteAccountDetails.accountId
Threat list name	service.additionalInfo.threatListName
Resource role	service.resourceRole
EKS cluster name	resource.eksClusterDetails.name
Kubernetes workload name	resource.kubernetesDetails.kubernetesWorkloadDetails.name
Kubernetes workload namespace	resource.kubernetesDetails.kubernetesWorkloadDetails.namespace
Kubernetes user name	resource.kubernetesDetails.kubernetesUserDetails.username
Kubernetes container image	resource.kubernetesDetails.kubernetesWorkloadDetails.containers.image
Kubernetes container image prefix	resource.kubernetesDetails.kubernetesWorkloadDetails.containers.imagePrefix
Scan ID	service.ebsVolumeScanDetails.scanId
EBS volume scan threat name	service.ebsVolumeScanDetails.scanDetections.threatDetectedByName.threatNames.name

Console field name	JSON field name
S3 object scan threat name	service.malwareScanDetails.threats.name
Threat severity	service.ebsVolumeScanDetails.scanDetections.threatDetectedByName.threatNames.severity
File SHA	service.ebsVolumeScanDetails.scanDetections.threatDetectedByName.threatNames.filePaths.hash
ECS cluster name	resource.ecsClusterDetails.name
ECS container image	resource.ecsClusterDetails.taskDetails.containers.image
ECS task definition ARN	resource.ecsClusterDetails.taskDetails.definitionArn
Standalone container image	resource.containerDetails.image
Database Instance Id	resource.rdsDbInstanceDetails.dbInstanceIdentifier
Database Cluster Id	resource.rdsDbInstanceDetails.dbClusterIdentifier
Database Engine	resource.rdsDbInstanceDetails.engine
Database user	resource.rdsDbUserDetails.user
Database instance tag key	resource.rdsDbInstanceDetails.tags.key
Database instance tag value	resource.rdsDbInstanceDetails.tags.value
Executable SHA-256	service.runtimeDetails.process.executableSha256
Process name	service.runtimeDetails.process.name

Console field name	JSON field name
Executable path	service.runtimeDetails.process.executablePath
Lambda function name	resource.lambdaDetails.functionName
Lambda function ARN	resource.lambdaDetails.functionArn
Lambda function tag key	resource.lambdaDetails.tags.key
Lambda function tag value	resource.lambdaDetails.tags.value
DNS request domain	service.action.dnsRequestAction.domainWithSuffix

Suppression rules in GuardDuty

A suppression rule is a set of criteria, consisting of a filter attribute paired with a value, used to filter findings by automatically archiving new findings that match the specified criteria. Suppression rules can be used to filter low-value findings, false positive findings, or threats you do not intend to act on, to make it easier to recognize the security threats with the most impact to your environment.

After you create a suppression rule, new findings that match the criteria defined in the rule are automatically archived as long as the suppression rule is in place. You can use an existing filter to create a suppression rule or create a suppression rule from a new filter you define. You can configure suppression rules to suppress entire finding types, or define more granular filter criteria to suppress only specific instances of a particular finding type. You can edit the suppression rules at any time.

Suppressed findings are not sent to AWS Security Hub, Amazon Simple Storage Service, Amazon Detective, or Amazon EventBridge, reducing finding noise level if you consume GuardDuty findings via Security Hub, a third-party SIEM, or other alerting and ticketing applications. If you've enabled Malware Protection for EC2, the suppressed GuardDuty findings won't initiate a malware scan.

GuardDuty continues to generate findings even when they match your suppression rules, however, those findings are automatically marked as **archived**. The archived finding is stored in GuardDuty for 90-days and can be viewed at any time during that period. You can view suppressed findings in the GuardDuty console by selecting **Archived** from the findings table, or through the GuardDuty

API using the ListFindings API with a `findingCriteria` criterion of `service.archived` equal to true.

Note

In a multi-account environment only the GuardDuty administrator can create suppression rules.

Using suppression rules with Extended Threat Detection

GuardDuty Extended Threat Detection automatically detects multi-stage attacks that span data sources, multiple types of AWS resources, and time, within an AWS account. It correlates events across different data sources to identify scenarios that present themselves as a potential threat to your AWS environment, and then generates an attack sequence finding. For more information, see How Extended Threat Detection works.

When you create suppression rules that archive findings, Extended Threat Detection can't use these archived findings when correlating events for attack sequences. Broad suppression rules might impact the ability of GuardDuty to detect behaviors aligned with detecting multi-stage attacks. Findings that are archived because of suppression rules are not considered as signals for attack sequences. For example, if you create a suppression rule that archives all EKS cluster-related findings instead of targeting specific known activities, GuardDuty won't be able to use those findings to detect an attack sequence where a threat actor exploits a container, obtains privileged tokens, and accesses sensitive resources.

Consider the following recommendations from GuardDuty:

- Continue using suppression rules to reduce alerts from known trusted activities.
- Keep the suppression rules focused on specific behaviors for which you don't want GuardDuty to generate a finding.

Common use cases for suppression rules and examples

The following finding types have common use cases for applying suppression rules. Select the finding name to learn more about that finding. Review the use case description to decide if you want to build a suppression rule for that finding type.

> **Important**
>
> GuardDuty recommends that you build suppression rules reactively and only for findings for which you have repeatedly identified false positives in your environment.

- UnauthorizedAccess:IAMUser/InstanceCredentialExfiltration.OutsideAWS – Use a suppression rule to automatically archive findings generated when VPC networking is configured to route internet traffic such that it egresses from an on-premises gateway rather than from a VPC Internet Gateway.

  This finding is generated when networking is configured to route internet traffic such that it egresses from an on-premises gateway rather than from a VPC Internet Gateway (IGW). Common configurations, such as using AWS Outposts, or VPC VPN connections, can result in traffic routed this way. If this is expected behavior, it is recommended that you use suppression rules and create a rule that consists of two filter criteria. The first criteria is **finding type**, which should be `UnauthorizedAccess:IAMUser/InstanceCredentialExfiltration.OutsideAWS`. The second filter criteria is **API caller IPv4 address** with the IP address or CIDR range of your on-premises internet gateway. The example below represents the filter you would use to suppress this finding type based on API caller IP address.

  ```
  Finding type: UnauthorizedAccess:IAMUser/InstanceCredentialExfiltration.OutsideAWS
   API caller IPv4 address: 198.51.100.6
  ```

  > **Note**
  >
  > To include multiple API caller IPs you can add a new API Caller IPv4 address filter for each.

- Recon:EC2/Portscan – Use a suppression rule to automatically archive findings when using a vulnerability assessment application.

  The suppression rule should consist of two filter criteria. The first criteria should use the **Finding type** attribute with a value of `Recon:EC2/Portscan`. The second filter criteria should match the instance or instances that host these vulnerability assessment tools. You can use either the **Instance image ID** attribute or the **Tag** value attribute depending on which criteria are identifiable with the instances that host these tools. The example below represents the filter you would use to suppress this finding type based on instances with a certain AMI.

```
Finding type: Recon:EC2/Portscan Instance image ID: ami-999999999
```

- UnauthorizedAccess:EC2/SSHBruteForce – Use a suppression rule to automatically archive findings when it is targeted to bastion instances.

  If the target of the brute force attempt is a bastion host, this may represent expected behavior for your AWS environment. If this is the case, we recommend that you set up a suppression rule for this finding. The suppression rule should consist of two filter criteria. The first criteria should use the **Finding type** attribute with a value of `UnauthorizedAccess:EC2/SSHBruteForce`. The second filter criteria should match the instance or instances that serve as a bastion host. You can use either the **Instance image ID** attribute or the **Tag** value attribute depending on which criteria is identifiable with the instances that host these tools. The example below represents the filter you would use to suppress this finding type based on instances with a certain instance tag value.

  ```
  Finding type: UnauthorizedAccess:EC2/SSHBruteForce Instance tag value: devops
  ```

- Recon:EC2/PortProbeUnprotectedPort – Use a suppression rule to automatically archive findings when it is targeted to intentionally exposed instances.

  There may be cases in which instances are intentionally exposed, for example if they are hosting web servers. If this is the case in your AWS environment, we recommend that you set up a suppression rule for this finding. The suppression rule should consist of two filter criteria. The first criteria should use the **Finding type** attribute with a value of `Recon:EC2/PortProbeUnprotectedPort`. The second filter criteria should match the instance or instances that serve as a bastion host. You can use either the **Instance image ID** attribute or the **Tag** value attribute, depending on which criteria is identifiable with the instances that host these tools. The example below represents the filter you would use to suppress this finding type based on instances with a certain instance tag key in the console.

  ```
  Finding type: Recon:EC2/PortProbeUnprotectedPort Instance tag key: prod
  ```

Recommended suppression rules for Runtime Monitoring findings

- PrivilegeEscalation:Runtime/DockerSocketAccessed gets generated when a process inside a container communicates with the Docker socket. There may be containers in your environment that may need to access the Docker socket for legitimate reasons. Access from such containers will generate PrivilegeEscalation:Runtime/DockerSocketAccessed finding. If this is a case in your AWS environment, we recommend that you set up a suppression rule for this finding type. The first criteria should use the **Finding type** field with value equal to `PrivilegeEscalation:Runtime/DockerSocketAccessed`. The second filter criteria is **Executable path** field with value equal to the process's `executablePath` in the generated finding. Alternatively, the second filter criteria can use **Executable SHA-256** field with value equal to the process's `executableSha256` in the generated finding.

- Kubernetes clusters run their own DNS servers as pods, such as `coredns`. Therefore, for each DNS lookup from a pod, GuardDuty captures two DNS events – one from the pod and the other from the server pod. This may generate duplicates for the following DNS findings:
  - Backdoor:Runtime/C&CActivity.B!DNS
  - CryptoCurrency:Runtime/BitcoinTool.B!DNS
  - Impact:Runtime/AbusedDomainRequest.Reputation
  - Impact:Runtime/BitcoinDomainRequest.Reputation
  - Impact:Runtime/MaliciousDomainRequest.Reputation
  - Impact:Runtime/SuspiciousDomainRequest.Reputation
  - Trojan:Runtime/BlackholeTraffic!DNS
  - Trojan:Runtime/DGADomainRequest.C!DNS
  - Trojan:Runtime/DriveBySourceTraffic!DNS
  - Trojan:Runtime/DropPoint!DNS
  - Trojan:Runtime/PhishingDomainRequest!DNS

  The duplicate findings will include pod, container, and process details that correspond to your DNS server pod. You may set up a suppression rule to suppress these duplicate findings using these fields. The first filter criteria should use the **Finding type** field with value equal to a DNS finding type from the list of findings provided earlier in this section. The second filter criteria could be either **Executable path** with value equal to your DNS server's `executablePath` or **Executable SHA-256** with value equal to your DNS server's `executableSHA256` in the generated finding. As an optional third filter criteria, you can use **Kubernetes container image** field with value equal to the container image of your DNS server pod in the generated finding.

Creating suppression rules in GuardDuty

A suppression rule is a set of criteria that includes using filter attributes and providing values for which you don't want GuardDuty to generate a finding type. The finding types that match this criteria are automatically archived. To reduce noise, the suppressed findings are not sent to any of the AWS services with which you may integrate. For more information about common use cases for creating suppression rules, see Suppression rules.

You can visualize, create, and manage suppression rules by using the GuardDuty console. Suppression rules are generated in the same manner as filters, and your existing saved filters can be used as suppression rules. For more information about creating filters, see Filtering findings in GuardDuty.

Choose your preferred access method to create a suppression rule for GuardDuty finding types.

Console

To create a suppression rule using the console:

1. Open the GuardDuty console at https://console.aws.amazon.com/guardduty/.
2. On the **Findings** page, the **Create suppression rule** feature remains grayed out unless you add at least one filter criterion. Because suppression rules are applied to active, ongoing findings, make sure that the **Status** menu is set to **Current**.
3. To add one or more filter criteria, follow steps 3 through 7 in Adding filters on Findings page, and then continue with the following steps.
4. After you have added the filter criteria and confirmed that the filtered findings meet your requirements, choose **Create suppression rule**.
5. Enter a **Name** for the suppression rule.The name must be 3-64 characters. Valid characters are a-z, A-Z, 0-9, period (.), hyphen (-), and underscore (_).
6. The **Description** is optional. If you enter a description, it can have up to 512 characters.
7. Choose **Create**.

You can also create a suppression rule from an existing saved filter. For more information about creating filters, see Filtering findings in GuardDuty.

To create a suppression rule from a saved filter:

1. Open the GuardDuty console at https://console.aws.amazon.com/guardduty/.

2. On the **Findings** page, from the **Saved rules** menu, select a saved filter set rule. This will automatically display the filter set and findings that match the criteria.
3. You can also add more filter criteria to this saved rule. If you don't need additional filter criteria, skip this step.

   To add one or more additional filter criteria, follow steps 2 through the end of the preceding procedure - To create a suppression rule using the console.
4. If you don't need to add additional filter criteria to the saved rule, follow steps 4 through the end of the preceding procedure - To create a suppression rule using the console.

API/CLI

To create a suppression rule using API:

1. You can create suppression rules through the CreateFilter API. To do so, specify the filter criteria in a JSON file following the format of the example detailed below. The below example will suppress any unarchived low-severity findings that has a DNS request to the `test.example.com` domain. For medium severity findings, the input list will be `["4", "5", "7"]`. For high severity findings, the input list will be `["6", "7", "8"]`. For critical severity findings, the input list will be `["9", "10"]`. You can also filter on the basis of any one value in the list.

   The following example adds a filter for low severity findings.

```
{
    "Criterion": {
        "service.archived": {
            "Eq": [
                "false"
            ]
        },
        "service.action.dnsRequestAction.domain": {
            "Eq": [
                "test.example.com"
            ]
        },
        "severity": {
            "Eq": [
                "1",
                "2",
```

```
                "3"
            ]
        }
    }
}
```

For a list of JSON field names and their console equivalent see Property filters in GuardDuty.

To test your filter criteria, use the same JSON criterion in the ListFindings API, and confirm that the correct findings have been selected. To test your filter criteria using AWS CLI follow the example using your own detectorId and .json file.

To find the `detectorId` for your account and current Region, see the **Settings** page in the https://console.aws.amazon.com/guardduty/ console, or run the ListDetectors API.

```
aws guardduty list-findings --detector-id 12abc34d567e8fa901bc2d34e56789f0 --
finding-criteria file://criteria.json
```

2. Upload your filter to be used as suppression rule with the CreateFilter API or by using the AWS CLI following the example below with your own detector ID, a name for the suppression rule, and .json file.

   To find the `detectorId` for your account and current Region, see the **Settings** page in the https://console.aws.amazon.com/guardduty/ console, or run the ListDetectors API.

```
aws guardduty create-filter --action ARCHIVE --detector-
id 12abc34d567e8fa901bc2d34e56789f0 --name yourfiltername --finding-criteria
 file://criteria.json
```

You can view a list of your filters programmatically with the ListFilter API. You can view the details of an individual filter by supplying the filter name to the GetFilter API. Update filters using UpdateFilter or delete them with the DeleteFilter API.

Deleting suppression rules in GuardDuty

This section provides the steps to delete a suppression rule in your AWS account in a specific AWS Region.

You may want to delete a suppression rule that no longer depicts an expected behavior in your environment. You no longer want to suppress the associated finding type so that GuardDuty can generate a finding type.

If you're a member account, your administrator account can take this action on your behalf. For more information, see Administrator account and member account relationships.

Choose your preferred access method to delete a suppression rule for GuardDuty finding types.

Console

1. Open the GuardDuty console at https://console.aws.amazon.com/guardduty/.
2. On the **Findings** page, choose **Suppress Findings** to open the suppression rule panel.
3. From the **Saved rules** drop down, choose a saved filter.
4. Choose **Delete rule**.

API/CLI

Run the DeleteFilter API. Specify the filter name and the associated detector ID for the particular Region.

Alternatively, you can use the following AWS CLI example by replacing the values formatted in *red*:

```
aws guardduty delete-filter --region us-east-1 --detector-id 12abc34d567e8fa901bc2d34e56789f0 --filter-name filterName
```

To find the `detectorId` for your account and current Region, see the **Settings** page in the https://console.aws.amazon.com/guardduty/ console, or run the ListDetectors API.

Customizing threat detection with entity lists and IP address lists

Amazon GuardDuty monitors the security of your AWS environment by analyzing and processing VPC Flow Logs, AWS CloudTrail event logs, and DNS logs. By enabling one or more Use-case focused GuardDuty protection plans (except Runtime Monitoring, you can expand the monitoring capabilities within GuardDuty.

With lists, GuardDuty helps you customize the scope of threat detection in your environment. You can configure GuardDuty to stop generating findings from your trusted sources and generate findings for known malicious sources from your threat lists. GuardDuty continues to support legacy IP address lists and extends support to entity lists (recommended) that can contain IP addresses, domains, or both.

Topics

- Understanding entity lists and IP address lists
- Important considerations for GuardDuty lists
- List formats
- Understanding list statuses
- Setting up prerequisites for entity lists and IP address lists
- Adding and activating an entity list or IP list
- Updating an entity list or IP address list
- De-activating entity list or IP address list
- Deleting entity list or IP address list

Understanding entity lists and IP address lists

GuardDuty offers two implementation approaches: entity lists (recommended) and IP lists. Both approaches help you specify trusted sources, which stop GuardDuty from generate findings and known threats, which GuardDuty uses to generate findings.

Entity lists support both IP addresses and domain names. They use direct Amazon Simple Storage Service (Amazon S3) access with a single IAM permission that doesn't impact IAM policy size limits across multiple Regions.

IP lists support only IP addresses and use [GuardDuty service-linked role (SLR)](#) (SLR), requiring IAM policy updates per Region, which may impact IAM policy size limits.

Trusted lists (both entity lists and IP address lists) include entries that you trust for secure communication with your AWS infrastructure. GuardDuty does not generate findings for entries listed in trusted sources. At any given time, you can add only one trusted entity list and one trusted IP address list per AWS account per Region.

Threat lists (both entity lists and IP address lists) include entries that you have identified as known malicious sources. When GuardDuty detects an activity involving these sources, it generates findings to alert you of potential security issues. You can create your own threat lists or incorporate third-party threat intelligence feeds. This list can be supplied by third-party threat intelligence or created specifically for your organization. In addition to generating findings because of a potentially suspicious activity, GuardDuty also generates findings based on an activity that involves entries from your threat lists. At any given time, you can upload up to six threat entity lists and threat IP address lists per AWS account per Region.

> **Note**
>
> To migrate from IP address lists to entity lists, follow [Prerequisites for entity lists](#), then add and activate the required entity list. After this, you can choose to deactivate or delete the corresponding IP address list.

Important considerations for GuardDuty lists

Before you begin working with lists, read the following considerations:

- IP address lists and entity lists apply only to traffic destined for publicly routable IP addresses and domains.
- In an entity list, the entries apply to CloudTrail, VPC Flow Logs in Amazon VPC, and Route53 Resolver DNS query logs findings.

  In an IP address list, the entries apply to CloudTrail and VPC Flow Logs in Amazon VPC findings, but not to Route53 Resolver DNS query logs findings.
- If you include the same IP address or domain in both trusted and threat lists, then this entry in the trusted list will take precedence. GuardDuty will not generate a finding if there is an activity associated with this entry.

- In a multi-account environment, only the GuardDuty administrator account can manage lists. This setting automatically applies to the member accounts. GuardDuty generates findings based on an activity that involves known malicious IP addresses (and domains) from the administrator account's threat sources, and doesn't generate findings based on activity that involves IP addresses (and domains) from the administrator account's trusted sources. For more information, see Multiple accounts in Amazon GuardDuty.
- Only IPv4 addresses are accepted. IPv6 addresses are not supported.
- After you activate, deactivate, or delete an entity list or IP address list, the process is estimated to complete within 15 minutes. In certain scenarios, it may take up to 40 minutes for this process to complete.
- GuardDuty uses a list for threat detection only when the status of the list becomes **Active**.
- Whenever you add or update an entry in the list's S3 bucket location, you must activate the list again. For more information, see Updating an entity list or IP address list.
- Entity lists and IP addresses have different quotas. For more information, see GuardDuty quotas.

List formats

GuardDuty accepts multiple file formats for your lists and entity lists, with a maximum of 35 MB per file. Each format has specific requirements and capabilities.

Plaintext (TXT)

This format supports IP addresses, CIDR ranges, and domain names. Each entry must appear on a separate line.

Example Example for entity list

```
192.0.2.1
192.0.2.0/24
example.com
example.org
*.example.org
```

Example Example for IP address list

```
192.0.2.0/24
```

```
198.51.100.1
203.0.113.1
```

Structured Threat Information Expression (STIX)

This format supports IP addresses, CIDR block, and domain names. STIX allows you to include additional context with your threat intelligence. GuardDuty processes IP addresses, CIDR ranges, and domain names from the STIX indicators.

Example Example for an entity list

```
<?xml version="1.0" encoding="UTF-8"?>
<stix:STIX_Package
    xmlns:cyboxCommon="http://cybox.mitre.org/common-2"
    xmlns:cybox="http://cybox.mitre.org/cybox-2"
    xmlns:cyboxVocabs="http://cybox.mitre.org/default_vocabularies-2"
    xmlns:stix="http://stix.mitre.org/stix-1"
    xmlns:indicator="http://stix.mitre.org/Indicator-2"
    xmlns:stixCommon="http://stix.mitre.org/common-1"
    xmlns:stixVocabs="http://stix.mitre.org/default_vocabularies-1"
    xmlns:DomainNameObj="http://cybox.mitre.org/objects#DomainNameObject-1"
    id="example:Package-a1b2c3d4-1111-2222-3333-444455556666"
    version="1.2">
    <stix:Indicators>
        <stix:Indicator
            id="example:indicator-a1b2c3d4-aaaa-bbbb-cccc-ddddeeeeffff"
            timestamp="2025-08-12T00:00:00Z"
            xsi:type="indicator:IndicatorType"
            xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
            <indicator:Title>Malicious domain observed Example</indicator:Title>
            <indicator:Type xsi:type="stixVocabs:IndicatorTypeVocab-1.1">Domain
 Watchlist</indicator:Type>
            <indicator:Observable id="example:Observable-0000-1111-2222-3333">
                <cybox:Object id="example:Object-0000-1111-2222-3333">
                    <cybox:Properties xsi:type="DomainNameObj:DomainNameObjectType">
                        <DomainNameObj:Value condition="Equals">bad.example.com</
DomainNameObj:Value>
                    </cybox:Properties>
                </cybox:Object>
            </indicator:Observable>
        </stix:Indicator>
    </stix:Indicators>
</stix:STIX_Package>
```

Example Example for an IP address list

```
<?xml version="1.0" encoding="UTF-8"?>
<stix:STIX_Package
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xmlns:stix="http://stix.mitre.org/stix-1"
    xmlns:stixCommon="http://stix.mitre.org/common-1"
    xmlns:ttp="http://stix.mitre.org/TTP-1"
    xmlns:cybox="http://cybox.mitre.org/cybox-2"
    xmlns:AddressObject="http://cybox.mitre.org/objects#AddressObject-2"
    xmlns:cyboxVocabs="http://cybox.mitre.org/default_vocabularies-2"
    xmlns:stixVocabs="http://stix.mitre.org/default_vocabularies-1"
    xmlns:example="http://example.com/"
    xsi:schemaLocation="
    http://stix.mitre.org/stix-1 http://stix.mitre.org/XMLSchema/core/1.2/
stix_core.xsd
    http://stix.mitre.org/Campaign-1 http://stix.mitre.org/XMLSchema/campaign/1.2/
campaign.xsd
    http://stix.mitre.org/Indicator-2 http://stix.mitre.org/XMLSchema/indicator/2.2/
indicator.xsd
    http://stix.mitre.org/TTP-2 http://stix.mitre.org/XMLSchema/ttp/1.2/ttp.xsd
    http://stix.mitre.org/default_vocabularies-1 http://stix.mitre.org/XMLSchema/
default_vocabularies/1.2.0/stix_default_vocabularies.xsd
    http://cybox.mitre.org/objects#AddressObject-2 http://cybox.mitre.org/XMLSchema/
objects/Address/2.1/Address_Object.xsd"
    id="example:STIXPackage-a78fc4e3-df94-42dd-a074-6de62babfe16"
    version="1.2">
    <stix:Observables cybox_major_version="1" cybox_minor_version="1">
        <cybox:Observable id="example:observable-80b26f43-
dc41-43ff-861d-19aff31e0236">
            <cybox:Object id="example:object-161a5438-1c26-4275-ba44-a35ba963c245">
                <cybox:Properties xsi:type="AddressObject:AddressObjectType"
 category="ipv4-addr">

 <AddressObject:Address_Valuecondition="InclusiveBetween">192.0.2.0##comma##192.0.2.255</
AddressObject:Address_Value>
                </cybox:Properties>
            </cybox:Object>
        </cybox:Observable>
        <cybox:Observable id="example:observable-b442b399-aea4-436f-bb34-
b9ef6c5ed8ab">
            <cybox:Object id="example:object-b422417f-bf78-4b34-ba2d-de4b09590a6d">
                <cybox:Properties xsi:type="AddressObject:AddressObjectType"
 category="ipv4-addr">
```

```
                    <AddressObject:Address_Value>198.51.100.1</
AddressObject:Address_Value>
                </cybox:Properties>
            </cybox:Object>
        </cybox:Observable>
        <cybox:Observable
 id="example:observable-1742fa06-8b5e-4449-9d89-6f9f32595784">
            <cybox:Object id="example:object-dc73b749-8a31-46be-803f-71df77565391">
                <cybox:Properties xsi:type="AddressObject:AddressObjectType"
 category="ipv4-addr">
                    <AddressObject:Address_Value>203.0.113.1</
AddressObject:Address_Value>
                </cybox:Properties>
            </cybox:Object>
        </cybox:Observable>
    </stix:Observables>
</stix:STIX_Package>
```

Open Threat Exchange (OTX)™ CSV

This format supports CIDR block, individual IP addresses, and domains. This file format has comma-separated values.

Example Example for entity list

```
Indicator type, Indicator, Description
CIDR, 192.0.2.0/24, example
IPv4, 198.51.100.1, example
IPv4, 203.0.113.1, example
Domain name, example.net, example
```

Example Example for IP address list

```
Indicator type, Indicator, Description
CIDR, 192.0.2.0/24, example
IPv4, 198.51.100.1, example
IPv4, 203.0.113.1, example
```

FireEye™ iSIGHT Threat Intelligence CSV

This format supports CIDR block, individual IP addresses, and domains. The following sample lists uses a `FireEye`™ CSV format.

Example Example for entity list

```
reportId, title, threatScape, audience, intelligenceType, publishDate, reportLink,
 webLink, emailIdentifier, senderAddress, senderName, sourceDomain, sourceIp, subject,
 recipient, emailLanguage, fileName, fileSize, fuzzyHash, fileIdentifier, md5, sha1,
 sha256, description, fileType, packer, userAgent, registry, fileCompilationDateTime,
 filePath, asn, cidr, domain, domainTimeOfLookup, networkIdentifier, ip, port,
 protocol, registrantEmail, registrantName, networkType, url, malwareFamily,
 malwareFamilyId, actor, actorId, observationTime

01-00000001, Example, Test, Operational, threat, 1494944400,
 https://www.example.com/report/01-00000001, https://www.example.com/
report/01-00000001, , , , , , , , , , , , , , , , , , , , , , , , 192.0.2.0/24, , ,
 Related, , , , , , network, , Ursnif, 21a14673-0d94-46d3-89ab-8281a0466099, , ,
 1494944400

01-00000002, Example, Test, Operational, threat, 1494944400,
 https://www.example.com/report/01-00000002, https://www.example.com/
report/01-00000002, , , , , , , , , , , , , , , , , , , , , , , , , , , Related,
 198.51.100.1, , , , , network, , Ursnif,
 12ab7bc4-62ed-49fa-99e3-14b92afc41bf, , ,1494944400

01-00000003, Example, Test, Operational, threat, 1494944400,
 https://www.example.com/report/01-00000003, https://www.example.com/
report/01-00000003, , , , , , , , , , , , , , , , , , , , , , , , , , , Related,
 203.0.113.1, , , , , network, , Ursnif, 8a78c3db-7bcb-40bc-a080-75bd35a2572d, , ,
 1494944400

 01-00000002, Malicious domain observed in test, Test, Operational, threat,
 1494944400, https://www.example.com/report/01-00000002,https://www.example.com/
report/01-00000002,,,,,,,,,,,,,,,,,,,,, 203.0.113.0/24, example.com,, Related,
 203.0.113.0, 8080, UDP,,, network,, Ursnif, fc13984c-c767-40c9-8329-f4c59557f73b,,,
 1494944400
```

Example Example for IP address list

```
reportId, title, threatScape, audience, intelligenceType, publishDate, reportLink,
 webLink, emailIdentifier, senderAddress, senderName, sourceDomain, sourceIp, subject,
 recipient, emailLanguage, fileName, fileSize, fuzzyHash, fileIdentifier, md5, sha1,
 sha256, description, fileType, packer, userAgent, registry, fileCompilationDateTime,
 filePath, asn, cidr, domain, domainTimeOfLookup, networkIdentifier, ip, port,
 protocol, registrantEmail, registrantName, networkType, url, malwareFamily,
 malwareFamilyId, actor, actorId, observationTime
```

```
01-00000001, Example, Test, Operational, threat, 1494944400,
 https://www.example.com/report/01-00000001, https://www.example.com/
report/01-00000001, , , , , , , , , , , , , , , , , , , , , , , , , 192.0.2.0/24, , ,
 Related, , , , , , network, , Ursnif, 21a14673-0d94-46d3-89ab-8281a0466099, , ,
 1494944400

01-00000002, Example, Test, Operational, threat, 1494944400,
 https://www.example.com/report/01-00000002, https://www.example.com/
report/01-00000002, , , , , , , , , , , , , , , , , , , , , , , , , , , Related,
 198.51.100.1, , , , , network, , Ursnif,
 12ab7bc4-62ed-49fa-99e3-14b92afc41bf, , ,1494944400

01-00000003, Example, Test, Operational, threat, 1494944400,
 https://www.example.com/report/01-00000003, https://www.example.com/
report/01-00000003, , , , , , , , , , , , , , , , , , , , , , , , , , , Related,
 203.0.113.1, , , , , network, , Ursnif, 8a78c3db-7bcb-40bc-a080-75bd35a2572d, , ,
 1494944400
```

Proofpoint™ ET Intelligence Feed CSV

In ProofPoint CSV format, you can add IP either addresses or domain names in a one list. The following sample list uses the `Proofpoint` CSV format. Providing value for the `ports` parameter is optional. When you don't provide it, leave a trailing comma (,) at the end.

Example Example for entity list

```
domain, category, score, first_seen, last_seen, ports (|)
198.51.100.1, 1, 100, 2000-01-01, 2000-01-01,
203.0.113.1, 1, 100, 2000-01-01, 2000-01-01, 80
```

Example Example for IP address list

```
ip, category, score, first_seen, last_seen, ports (|)
198.51.100.1, 1, 100, 2000-01-01, 2000-01-01,
203.0.113.1, 1, 100, 2000-01-01, 2000-01-01, 80
```

AlienVault™ Reputation Feed

The following sample list uses the `AlienVault` format.

Example Example for entity list

```
192.0.2.1#4#2#Malicious Host#KR##37.5111999512,126.974098206#3
192.0.2.2#4#2#Scanning Host#IN#Gurgaon#28.4666996002,77.0333023071#3
192.0.2.3#4#2##CN#Guangzhou#23.1166992188,113.25#3
www.test.org#4#2#Malicious Host#CA#Brossard#45.4673995972,-73.4832000732#3
www.example.com#4#2#Malicious Host#PL##52.2393989563,21.0361995697#3
```

Example Example for IP address list

```
198.51.100.1#4#2#Malicious Host#US##0.0,0.0#3
203.0.113.1#4#2#Malicious Host#US##0.0,0.0#3
```

Understanding list statuses

When you add an entity list or an IP address list, GuardDuty shows the status of that list. The **Status** column indicates whether the list is effective and if any action is required. The following list describes valid status values:

- **Active** – Indicates the list is currently in use for custom threat detection.
- **Inactive** – Indicates that the list is currently not in use. For GuardDuty to use this list for threat detection in your environment, see Step 3: Activating an entity list or IP address list in Updating an entity list or IP address list.

  When you update a list, the status automatically changes to **Inactive**. You must activate it again for GuardDuty to consider the latest version of the updated details.
- **Error** – Indicates that there is an issue with the list. Hover over the status to view the error details.
- **Activating** – Indicates that GuardDuty has initiated the process of activating the list. You can continue monitoring the status for this list. If there is no error, the status should update to **Active**. While the status remains **Activating**, you can't perform any action on this list. It might take a few minutes for the list status to change to **Active**.
- **Deactivating** – Indicates that GuardDuty has initiated the process of deactivating the list. You can continue monitoring the status for this list. If there is no error, the status should update to **Inactive**. While the status remains **Deactivating**, you can't perform any action on this list.
- **Delete Pending** – Indicates that the list is in the process of being deleted. While the status remains **Delete Pending**, you can't perform any action on this list.

Setting up prerequisites for entity lists and IP address lists

GuardDuty uses entity lists and IP address lists to customize threat detection in your AWS environment. Entity lists (recommended) support both IP addresses and domain names, while IP address lists support only IP addresses. Before you begin creating these lists, you must add the required permissions for the type of list that you want to use.

Prerequisites for entity lists

When you add entity lists, GuardDuty reads your trusted and threat intelligence lists from S3 buckets. The role you use to create entity lists must have the `s3:GetObject` permission for the S3 buckets contains these lists.

> **Note**
>
> In a multi-account environment, only the GuardDuty administrator account can manage lists, which automatically apply to member accounts.

If you don't already have the `s3:GetObject` permission for the S3 bucket location, then use the following example policy and replace *`amzn-s3-demo-bucket`* with your S3 bucket location.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "s3:GetObject",
            "Resource": "arn:aws:s3:::amzn-s3-demo-bucket/[object-key]"
        }
    ]
}
```

Prerequisites for IP address lists

Various IAM identities require special permissions to work with trusted IP lists and threat lists in GuardDuty. An identity with the attached AmazonGuardDutyFullAccess_v2 (recommended) managed policy can only rename and deactivate uploaded trusted IP lists and threat lists.

To grant various identities full access to working with trusted IP lists and threat lists (in addition to renaming and deactivating, this includes adding, activating, deleting, and updating the location or name of the lists), make sure that the following actions are present in the permissions policy attached to a user, group, or role:

```
{
    "Effect": "Allow",
    "Action": [
        "iam:PutRolePolicy",
        "iam:DeleteRolePolicy"
    ],
    "Resource": "arn:aws:iam::555555555555:role/aws-service-role/
guardduty.amazonaws.com/AWSServiceRoleForAmazonGuardDuty"
}
```

Important

These actions are not included in the `AmazonGuardDutyFullAccess` managed policy.

Using SSE-KMS encryption with entity lists and IP lists

GuardDuty supports SSE-AES256 and SSE-KMS encryption for your lists. SSE-C is not supported. For more information about encryption types for S3, see [Protecting data using server-side encryption](#).

Regardless of whether you use entity lists or IP lists, if you use SSE-KMS, then add the following statement to your AWS KMS key policy. Replace *123456789012* with your own account ID.

```
{
    "Sid": "AllowGuardDutyServiceRole",
    "Effect": "Allow",
    "Principal": {
    "AWS": "arn:aws:iam::123456789012:role/aws-service-role/guardduty.amazonaws.com/
AWSServiceRoleForAmazonGuardDuty"
    },
    "Action": "kms:Decrypt*",
    "Resource": "*"
}
```

Adding and activating an entity list or IP list

Entity lists and IP address lists help you customize the threat detection capabilities in GuardDuty. For more information about these lists, see Understanding entity lists and IP address lists.
To manage the trusted and threat intelligence data for your AWS environment, GuardDuty recommends using entity lists. Before you begin, see Setting up prerequisites for entity lists and IP address lists.

Choose one of the following access methods to add and activate a trusted entity list, threat entity list, trusted IP list, or a threat IP list.

Console

(Optional) step 1: Fetching location URL of your list

1. Open the Amazon S3 console at https://console.aws.amazon.com/s3/.
2. In the navigation pane, choose **Buckets**.
3. Choose the Amazon S3 bucket name that contains the specific list that you want to add.
4. Choose the object (list) name to view its details.
5. Under the **Properties** tab, copy the **S3 URI** for this object.

Step 2: Adding trusted or threat intelligence data

1. Open the GuardDuty console at https://console.aws.amazon.com/guardduty/.
2. In the navigation pane, choose **Lists**.
3. On the **Lists** page, choose **Entity lists** or **IP address lists** tab.
4. Based on your selected tab, choose to add a trusted list or a threat list.
5. In the dialog box to add either trusted or threat list, do the following steps:

    a. For **List name**, enter a name for your list.

       List naming constraints – The name of your list can include lowercase letters, uppercase letters, numbers, dash (-), and underscore (_).

       For an IP address list, the name of your list must be unique within an AWS account and Region.

b. For **Location**, provide the location where you have uploaded your list. If you don't already have it, see Step 1: Fetching location URL of your list.

Format of location URL

- https://s3.amazonaws.com/bucket.name/file.txt
- https://s3-aws-region.amazonaws.com/bucket.name/file.txt
- http://bucket.s3.amazonaws.com/file.txt
- http://bucket.s3-aws-region.amazonaws.com/file.txt
- s3://bucket.name/file.txt

c. (Optional) For **Expected bucket owner**, you can enter the AWS account ID that owns the Amazon S3 bucket specified in the **Location** field.

When you don't specify an AWS account ID owner, then GuardDuty behaves differently for entity lists and IP address lists. For entity lists, GuardDuty will validate that the current member account owns the S3 bucket specified in the **Location** field. For IP address lists, if you don't specify an AWS account ID owner, GuardDuty doesn't perform any validation.

If GuardDuty finds that this S3 bucket doesn't belong to the specified account ID, you will get an error at the time of activating the list.

d. Select the **I agree** check box.

e. Choose **Add list**. By default, the **Status** of the added list is **Inactive**. For the list to be effective, you must activate the list.

Step 3: Activating an entity list or IP address list

1. Open the GuardDuty console at https://console.aws.amazon.com/guardduty/.
2. In the navigation pane, choose **Lists**.
3. On the **Lists** page, select the tab in which you want to activate the list - **Entity lists** or **IP address lists**.
4. Select one list that you want to activate. This will enable the **Action** and **Edit** menu.
5. Choose **Action**, and then choose **Activate**.

API/CLI

To add and activate a trusted entity list

1. Run [CreateTrustedEntitySet](). Make sure to provide the `detectorId` of the member account for which you want to create this trusted entity list. To find the `detectorId` for your account and current Region, see the **Settings** page in the https://console.aws.amazon.com/guardduty/ console, or run the [ListDetectors]() API.

   List naming constraints – The name of your list can include lowercase letters, uppercase letters, numbers, dash (-), and underscore (_).

2. Alternatively, you can do this by running the following AWS Command Line Interface command:

```
aws guardduty create-trusted-entity-set \
--detector-id 12abc34d567e8fa901bc2d34e56789f0 \
--name "AnyOrganization ListEXAMPLE" \
--format TXT \
--location "https://s3.amazonaws.com/amzn-s3-demo-bucket/DOC-EXAMPLE-SOURCE-FILE.format" \
--activate
```

   Replace `detector-id` with the detector ID of the member account for which you will create the trusted entity list, and other placeholder values that are *shown in red*.

   If you don't want to activate this newly created list, then replace the parameter `--activate` with `--no-activate`.

   The `expected-bucket-owner` parameter is optional. Whether or not you specify the value for this parameter, GuardDuty validates that the AWS account ID associated with this `--detector-id` value owns the S3 bucket specified in the `--location` parameter. If GuardDuty finds that this S3 bucket doesn't belong to the specified account ID, you will get an error at the time of activating this list.

To add and activate threat entity lists

1. Run [CreateThreatEntitySet](). Make sure to provide the `detectorId` of the member account for which you want to create this threat entity list. To find the `detectorId` for your

account and current Region, see the **Settings** page in the https://console.aws.amazon.com/guardduty/ console, or run the ListDetectors API.

List naming constraints – The name of your list can include lowercase letters, uppercase letters, numbers, dash (-), and underscore (_).

2. Alternatively, you can do this by running the following AWS Command Line Interface command:

```
aws guardduty create-threat-entity-set \
--detector-id 12abc34d567e8fa901bc2d34e56789f0 \
--name "AnyOrganization ListEXAMPLE" \
--format TXT \
--location "https://s3.amazonaws.com/amzn-s3-demo-bucket/DOC-EXAMPLE-SOURCE-FILE.format" \
--activate
```

Replace `detector-id` with the detector ID of the member account for which you will create the trusted entity list, and other placeholder values that are *`shown in red`*.

If you don't want to activate this newly created list, then replace the parameter `--activate` with `--no-activate`.

The `expected-bucket-owner` parameter is optional. Whether or not you specify the value for this parameter, GuardDuty validates that the AWS account ID associated with this `--detector-id` value owns the S3 bucket specified in the `--location` parameter. If GuardDuty finds that this S3 bucket doesn't belong to the specified account ID, you will get an error at the time of activating this list.

To add and activate a trusted IP address list

1. Run CreateIPSet. Make sure to provide the `detectorId` of the member account for which you want to create this trusted IP address list. To find the `detectorId` for your account and current Region, see the **Settings** page in the https://console.aws.amazon.com/guardduty/ console, or run the ListDetectors API.

   For an IP address list, the name of your list must be unique within an AWS account and Region.

List naming constraints – The name of your list can include lowercase letters, uppercase letters, numbers, dash (-), and underscore (_).

2. Alternatively, you can do this by running the following AWS Command Line Interface command and make sure to replace the `detector-id` with the detector ID of the member account for which you will update the trusted IP address list.

```
aws guardduty create-ip-set \
--detector-id 12abc34d567e8fa901bc2d34e56789f0 \
--name "AnyOrganization ListEXAMPLE" \
--format TXT \
--location "https://s3.amazonaws.com/amzn-s3-demo-bucket/DOC-EXAMPLE-SOURCE-FILE.format" \
--activate
```

Replace `detector-id` with the detector ID of the member account for which you will create the trusted IP list, and other placeholder values that are *`shown in red`*.

If you don't want to activate this newly created list, then replace the parameter `--activate` with `--no-activate`.

The `expected-bucket-owner` parameter is optional. When you don't specify the account ID that owns the S3 bucket, GuardDuty doesn't perform any validation. When you specify the account ID for the `expected-bucket-owner` parameter, GuardDuty validates that this AWS account ID owns the S3 bucket specified in the `--location` parameter. If GuardDuty finds that this S3 bucket doesn't belong to the specified account ID, you will get an error at the time of activating this list.

To add and activate threat IP lists

1. Run CreateThreatIntelSet. Make sure to provide the `detectorId` of the member account for which you want to create this threat IP address list. To find the `detectorId` for your account and current Region, see the **Settings** page in the https://console.aws.amazon.com/guardduty/ console, or run the ListDetectors API.

   List naming constraints – The name of your list can include lowercase letters, uppercase letters, numbers, dash (-), and underscore (_).

For an IP address list, the name of your list must be unique within an AWS account and Region.

2. Alternatively, you can do this by running the following AWS Command Line Interface command and make sure to replace the `detector-id` with the detector ID of the member account for which you will update the threat IP list.

```
aws guardduty create-threat-intel-set \
--detector-id 12abc34d567e8fa901bc2d34e56789f0 \
--name "AnyOrganization ListEXAMPLE" \
--format TXT \
--location "https://s3.amazonaws.com/amzn-s3-demo-bucket/DOC-EXAMPLE-SOURCE-FILE.format" \
--activate
```

Replace `detector-id` with the detector ID of the member account for which you will create the threat IP list, and other placeholder values that are *shown in red*.

If you don't want to activate this newly created list, then replace the parameter `--activate` with `--no-activate`.

The `expected-bucket-owner` parameter is optional. When you don't specify the account ID that owns the S3 bucket, GuardDuty doesn't perform any validation. When you specify the account ID for the `expected-bucket-owner` parameter, GuardDuty validates that this AWS account ID owns the S3 bucket specified in the `--location` parameter. If GuardDuty finds that this S3 bucket doesn't belong to the specified account ID, you will get an error at the time of activating this list.

After you activate an entity list or IP address list, it might take a few minutes for this list to be effective. For more information, see Important considerations for GuardDuty lists.

Updating an entity list or IP address list

Entity lists and IP address lists help you customize the threat detection capabilities in GuardDuty. For more information about these lists, see Understanding entity lists and IP address lists.

You can update the name of a list, S3 bucket location, expected bucket owner account ID, and the entries in an existing list. If you update the entries in a list, you must follow the steps to activate the list again for GuardDuty to use the latest version of the list. After you update or activate an entity list or IP address list, it might take a few minutes for this list to be effective. For more information, see Important considerations for GuardDuty lists.

> **Note**
>
> If the status of a list is **Activating**, **Deactivating**, or **Delete Pending**, you must wait for a few minutes before performing any action. For information about these statuses, see Understanding list statuses.

Choose one of the access methods to update an entity list or IP address list.

Console

1. Open the GuardDuty console at https://console.aws.amazon.com/guardduty/.
2. In the navigation pane, choose **Lists**.
3. On the **Lists** page, select the appropriate tab - **Entity lists** or **IP address lists**.
4. Select one list (trusted or threat) that you want to update. This will enable the **Action** and **Edit** menu.
5. Choose **Edit**.
6. In the dialog box to update the list, specify the details that you want to update.

   List naming constraints – The name of your list can include lowercase letters, uppercase letters, numbers, dash (-), and underscore (_).

   For an IP address list, the name of your list must be unique within an AWS account and Region.
7. (Optional) For **Expected bucket owner**, you can enter the AWS account ID that owns the Amazon S3 bucket specified in the **Location** field.

   When you don't specify an AWS account ID owner, then GuardDuty behaves differently for entity lists and IP address lists. For entity lists, GuardDuty will validate that the current member account owns the S3 bucket specified in the **Location** field. For IP address lists, if you don't specify an AWS account ID owner, GuardDuty doesn't perform any validation.

   If GuardDuty finds that this S3 bucket doesn't belong to the specified account ID, you will get an error at the time of activating the list.

8. Choose the **I agree** check box, and then choose **Update list**. The value in the **Status** column will change to **Inactive**.

9. **Activating the updated list**

   a. In the selected tab (**Entity lists** or **IP address lists**), select the list that you want to activate.

   b. Choose **Actions**, and then choose **Activate**.

API/CLI

To begin with the following procedures, you need the ID, such as `trustedEntitySetId`, `threatEntitySetId`, `trustedIpSet`, or `threatIpSet`, that is associated with the list resource you want to update.

To update and activate a trusted entity list

1. Run [UpdateTrustedEntitySet](). Make sure to provide the `detectorId` of the member account for which you want to update this trusted entity list. To find the `detectorId` for your account and current Region, see the **Settings** page in the [https://console.aws.amazon.com/guardduty/]() console, or run the [ListDetectors]() API.

   List naming constraints – The name of your list can include lowercase letters, uppercase letters, numbers, dash (-), and underscore (_).

2. Alternatively, you can do this by running the following AWS Command Line Interface command that updates the `name` of the list and also activates this list:

```
aws guardduty update-trusted-entity-set \
--detector-id 12abc34d567e8fa901bc2d34e56789f0 \
--name "AnyOrganization ListEXAMPLE" \
--trusted-entity-set-id d4b94fc952d6912b8f3060768example \
--activate
```

   Replace `detector-id` with the detector ID of the member account for which you will create the trusted entity list, and other placeholder values that are *`shown in red`*.

   If you don't want to activate this newly created list, then replace the parameter `--activate` with `--no-activate`.

The `expected-bucket-owner` parameter is optional. Whether or not you specify the value for this parameter, GuardDuty validates that the AWS account ID associated with this `--detector-id` value owns the S3 bucket specified in the `--location` parameter. If GuardDuty finds that this S3 bucket doesn't belong to the specified account ID, you will get an error at the time of activating this list.

To update and activate a threat entity list

1. Run UpdateThreatEntitySet. Make sure to provide the `detectorId` of the member account for which you want to create this threat entity list. To find the `detectorId` for your account and current Region, see the **Settings** page in the https://console.aws.amazon.com/guardduty/ console, or run the ListDetectors API.

   List naming constraints – The name of your list can include lowercase letters, uppercase letters, numbers, dash (-), and underscore (_).

2. Alternatively, you can do this by running the following AWS Command Line Interface command that updates the `name` of the list and also activates this list:

   ```
   aws guardduty update-threat-entity-set \
   --detector-id 12abc34d567e8fa901bc2d34e56789f0 \
   --name "AnyOrganization ListEXAMPLE" \
   --threat-entity-set-id d4b94fc952d6912b8f3060768example \
   --activate
   ```

   Replace `detector-id` with the detector ID of the member account for which you will create the threat entity list, and other placeholder values that are *`shown in red`*.

   If you don't want to activate this newly created list, then replace the parameter `--activate` with `--no-activate`.

   The `expected-bucket-owner` parameter is optional. Whether or not you specify the value for this parameter, GuardDuty validates that the AWS account ID associated with this `--detector-id` value owns the S3 bucket specified in the `--location` parameter. If GuardDuty finds that this S3 bucket doesn't belong to the specified account ID, you will get an error at the time of activating this list.

To update and activate a trusted IP address list

1. Run [CreateIPSet](). Make sure to provide the `detectorId` of the member account for which you want to update this trusted IP address list. To find the `detectorId` for your account and current Region, see the **Settings** page in the [https://console.aws.amazon.com/guardduty/]() console, or run the [ListDetectors]() API.

   List naming constraints – The name of your list can include lowercase letters, uppercase letters, numbers, dash (-), and underscore (_).

   For an IP address list, the name of your list must be unique within an AWS account and Region.

2. Alternatively, you can do this by running the following AWS Command Line Interface command that also activates the list:

   ```
   aws guardduty update-ip-set \
   --detector-id 12abc34d567e8fa901bc2d34e56789f0 \
   --name "AnyOrganization ListEXAMPLE" \
   --ip-set-id d4b94fc952d6912b8f3060768example \
   --activate
   ```

   Replace `detector-id` with the detector ID of the member account for which you will update the trusted IP list, and other placeholder values that are *`shown in red`*.

   If you don't want to activate this newly created list, then replace the parameter `--activate` with `--no-activate`.

   The `expected-bucket-owner` parameter is optional. When you don't specify the account ID that owns the S3 bucket, GuardDuty doesn't perform any validation. When you specify the account ID for the `expected-bucket-owner` parameter, GuardDuty validates that this AWS account ID owns the S3 bucket specified in the `--location` parameter. If GuardDuty finds that this S3 bucket doesn't belong to the specified account ID, you will get an error at the time of activating this list.

To add and activate threat IP lists

1. Run [CreateThreatIntelSet](). Make sure to provide the `detectorId` of the member account for which you want to create this threat IP address list. To find the `detectorId` for your

account and current Region, see the **Settings** page in the https://console.aws.amazon.com/guardduty/ console, or run the ListDetectors API.

List naming constraints – The name of your list can include lowercase letters, uppercase letters, numbers, dash (-), and underscore (_).

For an IP address list, the name of your list must be unique within an AWS account and Region.

2. Alternatively, you can do this by running the following AWS Command Line Interface command that also activates the list:

```
aws guardduty update-threat-intel-set \
--detector-id 12abc34d567e8fa901bc2d34e56789f0 \
--name "AnyOrganization ListEXAMPLE" \
--threat-intel-set-id d4b94fc952d6912b8f3060768example \
--activate
```

Replace `detector-id` with the detector ID of the member account for which you will update the threat IP list, and other placeholder values that are *`shown in red`*.

If you don't want to activate this newly created list, then replace the parameter `--activate` with `--no-activate`.

The `expected-bucket-owner` parameter is optional. When you don't specify the account ID that owns the S3 bucket, GuardDuty doesn't perform any validation. When you specify the account ID for the `expected-bucket-owner` parameter, GuardDuty validates that this AWS account ID owns the S3 bucket specified in the `--location` parameter. If GuardDuty finds that this S3 bucket doesn't belong to the specified account ID, you will get an error at the time of activating this list.

De-activating entity list or IP address list

When you no longer want GuardDuty to use a list, you can deactivate it. It might take a few minutes for the process to complete. For more information, see Important considerations for GuardDuty lists. After the list gets deactivated, the entries in the entity list or IP address list will not impact threat detection in GuardDuty.

Choose one of the access methods to deactivate the list.

Console

To deactivate entity list or IP address list

1. Open the GuardDuty console at https://console.aws.amazon.com/guardduty/.
2. In the navigation pane, choose **Lists**.
3. On the **List** page, select the tab in which you want to deactivate the list - **Entity lists** or **IP address list**.
4. In the selected tab, select the list that you want to deactivate.
5. Choose **Actions**, and then choose **Deactivate**.
6. Confirm the action and choose **Deactivate**.

API/CLI

To begin with the following procedures, you need the ID, such as `trustedEntitySetId`, `threatEntitySetId`, `trustedIpSet`, or `threatIpSet`, that is associated with the list resource you want to deactivate.

To deactivate a trusted entity list

1. Run UpdateTrustedEntitySet. Make sure to provide the `detectorId` of the member account for which you want to deactivate this trusted entity list. To find the `detectorId` for your account and current Region, see the **Settings** page in the https://console.aws.amazon.com/guardduty/ console, or run the ListDetectors API.
2. Alternatively, you can do this by running the following AWS Command Line Interface command:

```
aws guardduty update-trusted-entity-set \
--detector-id 12abc34d567e8fa901bc2d34e56789f0 \
--trusted-entity-set-id d4b94fc952d6912b8f3060768example \
--no-activate
```

   Replace `detector-id` with the detector ID of the member account for which you will deactivate the trusted entity list, and other placeholder values that are *`shown in red`*.

To deactivate threat entity lists

1. Run UpdateThreatEntitySet. Make sure to provide the `detectorId` of the member account for which you want to deactivate this threat entity list. To find the `detectorId` for your account and current Region, see the **Settings** page in the https://console.aws.amazon.com/guardduty/ console, or run the ListDetectors API.
2. Alternatively, you can do this by running the following AWS Command Line Interface command:

```
aws guardduty update-threat-entity-set \
--detector-id 12abc34d567e8fa901bc2d34e56789f0 \
--threat-entity-set-id d4b94fc952d6912b8f3060768example \
--no-activate
```

   Replace `detector-id` with the detector ID of the member account for which you will create the threat entity list, and other placeholder values that are *`shown in red`*.

To deactivate a trusted IP address list

1. Run UpdateIPSet. Make sure to provide the `detectorId` of the member account for which you want to deactivate this trusted IP address list. To find the `detectorId` for your account and current Region, see the **Settings** page in the https://console.aws.amazon.com/guardduty/ console, or run the ListDetectors API.
2. Alternatively, you can do this by running the following AWS Command Line Interface command and make sure to replace the `detector-id` with the detector ID of the member account for which you will deactivate the trusted IP address list.

```
aws guardduty update-ip-set \
--detector-id 12abc34d567e8fa901bc2d34e56789f0 \
--ip-set-id d4b94fc952d6912b8f3060768example \
--no-activate
```

To deactivate threat IP list

1. Run UpdateThreatIntelSet. Make sure to provide the `detectorId` of the member account for which you want to deactivate this threat IP address list. To find the

`detectorId` for your account and current Region, see the **Settings** page in the https://console.aws.amazon.com/guardduty/ console, or run the ListDetectors API.

2. Alternatively, you can do this by running the following AWS Command Line Interface command and make sure to replace the `detector-id` with the detector ID of the member account for which you will deactivate the threat IP list.

```
aws guardduty update-threat-intel-set \
--detector-id 12abc34d567e8fa901bc2d34e56789f0 \
--threat-intel-set-id d4b94fc952d6912b8f3060768example \
--no-activate
```

Deleting entity list or IP address list

When you no longer want to keep a list entry in your entity set or IP address set, you can delete it. It might take a few minutes for the process to complete. For more information, see Important considerations for GuardDuty lists.

If the status of the list is **Activating** or **Deactivating**, you must wait for a few minutes before performing any action. For more information, see Understanding list statuses.

Choose one of the access methods to delete the list.

Console

To delete entity list or IP address list

1. Open the GuardDuty console at https://console.aws.amazon.com/guardduty/.
2. In the navigation pane, choose **Lists**.
3. On the **List** page, select the tab in which you want to delete the list - **Entity lists** or **IP address list**.
4. In the selected tab, select the list that you want to delete.
5. Choose **Actions**, and then choose **Delete**.

   The list status will change to **Delete Pending**. It might take a few minutes for the list to get deleted.

API/CLI

To begin with the following procedures, you need the ID, such as `trustedEntitySetId`, `threatEntitySetId`, `trustedIpSet`, or `threatIpSet`, that is associated with the list resource you want to delete.

To delete a trusted entity list

1. Run DeleteTrustedEntitySet. Make sure to provide the `detectorId` of the member account for which you want to delete this trusted entity list. To find the `detectorId` for your account and current Region, see the **Settings** page in the https://console.aws.amazon.com/guardduty/ console, or run the ListDetectors API.
2. Alternatively, you can do this by running the following AWS Command Line Interface command:

```
aws guardduty delete-trusted-entity-set \
--detector-id 12abc34d567e8fa901bc2d34e56789f0 \
--trusted-entity-set-id d4b94fc952d6912b8f3060768example
```

   Replace `detector-id` with the detector ID of the member account for which you will delete the trusted entity list, and other placeholder values that are *`shown in red`*.

To deactivate threat entity lists

1. Run DeleteThreatEntitySet. Make sure to provide the `detectorId` of the member account for which you want to delete this threat entity list. To find the `detectorId` for your account and current Region, see the **Settings** page in the https://console.aws.amazon.com/guardduty/ console, or run the ListDetectors API.
2. Alternatively, you can do this by running the following AWS Command Line Interface command:

```
aws guardduty delete-threat-entity-set \
--detector-id 12abc34d567e8fa901bc2d34e56789f0 \
--threat-entity-set-id d4b94fc952d6912b8f3060768example
```

   Replace `detector-id` with the detector ID of the member account for which you will delete the threat entity list, and other placeholder values that are *`shown in red`*.

To delete a trusted IP address list

1. Run [DeleteIPSet](). Make sure to provide the `detectorId` of the member account for which you want to delete this trusted IP address list. To find the `detectorId` for your account and current Region, see the **Settings** page in the https://console.aws.amazon.com/guardduty/ console, or run the [ListDetectors]() API.
2. Alternatively, you can do this by running the following AWS Command Line Interface command and make sure to replace the `detector-id` with the detector ID of the member account for which you will delete the trusted IP address list.

```
aws guardduty delete-ip-set \
--detector-id 12abc34d567e8fa901bc2d34e56789f0 \
--ip-set-id d4b94fc952d6912b8f3060768example
```

Replace `detector-id` with the detector ID of the member account for which you will delete the threat entity list, and other placeholder values that are *`shown in red`*.

To delete threat IP list

1. Run [DeleteThreatIntelSet](). Make sure to provide the `detectorId` of the member account for which you want to delete this threat IP address list. To find the `detectorId` for your account and current Region, see the **Settings** page in the https://console.aws.amazon.com/guardduty/ console, or run the [ListDetectors]() API.
2. Alternatively, you can do this by running the following AWS Command Line Interface command and make sure to replace the `detector-id` with the detector ID of the member account for which you will delete the threat IP list.

```
aws guardduty delete-threat-intel-set \
--detector-id 12abc34d567e8fa901bc2d34e56789f0 \
--threat-intel-set-id d4b94fc952d6912b8f3060768example
```

Replace `detector-id` with the detector ID of the member account for which you will delete the threat entity list, and other placeholder values that are *`shown in red`*.

Exporting generated GuardDuty findings to Amazon S3 buckets

GuardDuty retains the generated findings for a period of 90 days. GuardDuty exports the active findings to Amazon EventBridge (EventBridge). You can optionally export the generated findings to an Amazon Simple Storage Service (Amazon S3) bucket. This will help you to track the historical data of potentially suspicious activities in your account and evaluate whether the recommended remediation steps were successful.

Any new active findings that GuardDuty generates are automatically exported within about 5 minutes after the finding is generated. You can set the frequency for how often updates to the active findings are exported to EventBridge. The frequency that you select applies to the exporting of new occurrences of existing findings to EventBridge, your S3 bucket (when configured), and Detective (when integrated). For information about how GuardDuty aggregates multiple occurrences of existing findings, see GuardDuty finding aggregation.

When you configure settings to export findings to an Amazon S3 bucket, GuardDuty uses AWS Key Management Service (AWS KMS) to encrypt the findings data in your S3 bucket. This requires you to add permissions to your S3 bucket and the AWS KMS key so that GuardDuty can use them to export findings in your account.

Contents

- Considerations
- Step 1 – Permissions required to export findings
- Step 2 – Attaching policy to your KMS key
- Step 3 – Attaching policy to Amazon S3 bucket
- Step 4 - Exporting findings to an S3 bucket (Console)
- Step 5 – Setting frequency to export updated active findings

Considerations

Before proceeding with the prerequisites and steps to export findings, consider the following key concepts:

- **Export settings are regional** – You need to configure export options in each Region where you use GuardDuty.

- **Exporting findings to Amazon S3 buckets in different AWS Regions (cross-Region) –** GuardDuty supports the following export settings:
  - Your Amazon S3 bucket or object, and AWS KMS key must belong to the same AWS Region.
  - For the findings generated in a commercial Region, you can choose to export these findings to an S3 bucket in any commercial Region. However, you can't export these findings to an S3 bucket in an opt-in Region.
  - For the findings generated in an opt-in Region, you can choose to export these findings to the same opt-in Region where they're generated or any commercial Region. However, you can't export findings from one opt-in Region to another opt-in Region.
- **Permissions to export findings –** To configure settings for exporting active findings, your S3 bucket must have permissions that allows GuardDuty to upload objects. You must also have an AWS KMS key that GuardDuty can use to encrypt findings.
- **Archived findings are not exported –** The default behavior is that the archived findings, including new instances of suppressed findings, are not exported.

  When a GuardDuty finding gets generated as *Archived*, you will need to *Unarchive* it. This changes the **Filter finding status** to **Active**. GuardDuty exports the updates to the existing unarchived findings based on how you configure Step 5 – Frequency for exporting findings.
- **GuardDuty administrator account can export findings generated in associated member accounts –** When you configure export findings in an administrator account, all the findings from the associated member accounts that are generated in the same Region are also exported to the same location that you configured for the administrator account. For more information, see Understanding the relationship between GuardDuty administrator account and member accounts.

Step 1 – Permissions required to export findings

When you configure settings for exporting findings, you select an Amazon S3 bucket where you can store the findings and an AWS KMS key to use for data encryption. In addition to permissions for GuardDuty actions, you must also have permissions to the following actions to successfully configure settings to export findings:

- `s3:GetBucketLocation`
- `s3:PutObject`

If you need to export the findings to a specific prefix in your Amazon S3 bucket, you must also add the following permissions to the IAM role:

- `s3:GetObject`
- `s3:ListBucket`

Step 2 – Attaching policy to your KMS key

GuardDuty encrypts the findings data in your bucket by using AWS Key Management Service. To successfully configure the settings, you must first give GuardDuty permission to use a KMS key. You can grant the permissions by attaching the policy to your KMS key.

When you use a KMS key from another account, you need to apply the key policy by logging in to the AWS account that owns the key. When you configure the settings to export findings, you'll also need the key ARN from the account that owns the key.

To modify the KMS key policy for GuardDuty to encrypt your exported findings

1. Open the AWS KMS console at https://console.aws.amazon.com/kms.
2. To change the AWS Region, use the Region selector in the upper-right corner of the page.
3. Select an existing KMS key or perform the steps to Create a new key in the *AWS Key Management Service Developer Guide*, that you will use to encrypt the exported findings.

   > **Note**
   >
   > The AWS Region of your KMS key and the Amazon S3 bucket must be the same.

   You can use the same S3 bucket and KMS key pair to export the findings from any applicable Region. For more information, see Considerations for exporting findings across Regions.
4. In the **Key policy** section, choose **Edit**.

   If **Switch to policy view** is displayed, choose it to display the **Key policy**, and then choose **Edit**.
5. Copy the following policy block to your KMS key policy, to grant GuardDuty permission to use your key.

```
{
    "Sid": "AllowGuardDutyKey",
```

```
    "Effect": "Allow",
    "Principal": {
        "Service": "guardduty.amazonaws.com"
    },
    "Action": "kms:GenerateDataKey",
    "Resource": "KMS key ARN",
    "Condition": {
        "StringEquals": {
            "aws:SourceAccount": "123456789012",
            "aws:SourceArn":
 "arn:aws:guardduty:Region2:123456789012:detector/SourceDetectorID"
        }
    }
}
```

6. Edit the policy by replacing the following values that are formatted in *red* in the policy example:

   1. Replace *KMS key ARN* with the Amazon Resource Name (ARN) of the KMS key. To locate the key ARN, see Finding the key ID and ARN in the *AWS Key Management Service Developer Guide*.
   2. Replace *123456789012* with the AWS account ID that owns the GuardDuty account exporting the findings.
   3. Replace *Region2* with the AWS Region where the GuardDuty findings are generated.
   4. Replace *SourceDetectorID* with the `detectorID` of the GuardDuty account in the specific Region where the findings generated.

      To find the `detectorId` for your account and current Region, see the **Settings** page in the https://console.aws.amazon.com/guardduty/ console, or run the ListDetectors API.

> **Note**
>
> If you're using GuardDuty in an opt-in Region, replace the value for the "Service" with the Regional endpoint for that Region. For example, if you're using GuardDuty in the Middle East (Bahrain) (me-south-1) Region, replace `"Service": "guardduty.amazonaws.com"` with `"Service": "guardduty.me-south-1.amazonaws.com"`. For information about endpoints for each opt-in Region, see GuardDuty endpoints and quotas.

7. If you added the policy statement before the final statement, add a comma before adding this statement. Make sure that the JSON syntax of your KMS key policy is valid.

   Choose **Save**.

8. (Optional) copy the key ARN to a notepad for use in the later steps.

Step 3 – Attaching policy to Amazon S3 bucket

Add permissions to the Amazon S3 bucket to which you will export findings so that GuardDuty can upload objects to this S3 bucket. Independent of using an Amazon S3 bucket that belongs to either your account or in a different AWS account, you must add these permissions.

If at any point in time, you decide to export findings to a different S3 bucket, then to continue exporting findings, you must add permissions to that S3 bucket and configure the export findings settings again.

If you do not already have an Amazon S3 bucket where you want to export these findings, see Creating a bucket in the *Amazon S3 User Guide*.

To attach permissions to your S3 bucket policy

1. Perform the steps under To create or edit a bucket policy in the *Amazon S3 User Guide*, until the **Edit bucket policy** page appears.
2. The **example policy** shows how grant GuardDuty permission to export findings to your Amazon S3 bucket. If you change the path after you configure export findings, then you must modify the policy to grant permission to the new location.

   Copy the following **example policy** and paste it into the **Bucket policy editor**.

   If you added the policy statement before the final statement, add a comma before adding this statement. Make sure that the JSON syntax of your KMS key policy is valid.

   S3 bucket example policy

   JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
```

```
        {
            "Sid": "Allow GetBucketLocation",
            "Effect": "Allow",
            "Principal": {
                "Service": "guardduty.amazonaws.com"
            },
            "Action": "s3:GetBucketLocation",
            "Resource": "arn:aws:s3:::amzn-s3-demo-bucket",
            "Condition": {
                "StringEquals": {
                    "aws:SourceAccount": "123456789012",
                    "aws:SourceArn": "arn:aws:guardduty:us-
east-2:123456789012:detector/SourceDetectorID"

                }
            }
        },
        {
            "Sid": "Allow PutObject",
            "Effect": "Allow",
            "Principal": {
                "Service": "guardduty.amazonaws.com"
            },
            "Action": "s3:PutObject",
            "Resource": "arn:aws:s3:::amzn-s3-demo-bucket[optional prefix]/
*",
            "Condition": {
                "StringEquals": {
                    "aws:SourceAccount": "123456789012",
                    "aws:SourceArn": "arn:aws:guardduty:us-
east-2:123456789012:detector/SourceDetectorID"

                }
            }
        },
        {
            "Sid": "Deny unencrypted object uploads",
            "Effect": "Deny",
            "Principal": {
                "Service": "guardduty.amazonaws.com"
            },
            "Action": "s3:PutObject",
            "Resource": "arn:aws:s3:::amzn-s3-demo-bucket[optional prefix]/
*",
```

```
            "Condition": {
                "StringNotEquals": {
                    "s3:x-amz-server-side-encryption": "aws:kms"
                }
            }
        },
        {
            "Sid": "Deny incorrect encryption header",
            "Effect": "Deny",
            "Principal": {
                "Service": "guardduty.amazonaws.com"
            },
            "Action": "s3:PutObject",
            "Resource": "arn:aws:s3:::amzn-s3-demo-bucket[optional prefix]/
*",
            "Condition": {
                "StringNotEquals": {
                "s3:x-amz-server-side-encryption-aws-kms-key-id":
 "arn:aws:kms:us-east-2:111122223333:key/a1b2c3d4-5678-90ab-cdef-
EXAMPLE11111"
                }
            }
        },
        {
            "Sid": "Deny non-HTTPS access",
            "Effect": "Deny",
            "Principal": "*",
            "Action": "s3:*",
            "Resource": "arn:aws:s3:::amzn-s3-demo-bucket[optional prefix]/
*",
            "Condition": {
                "Bool": {
                    "aws:SecureTransport": "false"
                }
            }
        }
    ]
}
```

3. Edit the policy by replacing the following values that are formatted in *red* in the policy example:

1. Replace *`Amazon S3 bucket ARN`* with the Amazon Resource Name (ARN) of the Amazon S3 bucket. You can find the **Bucket ARN** on the **Edit bucket policy** page in the https://console.aws.amazon.com/s3/ console.
2. Replace *`123456789012`* with the AWS account ID that owns the GuardDuty account exporting the findings.
3. Replace *`Region2`* with the AWS Region where the GuardDuty findings are generated.
4. Replace *`SourceDetectorID`* with the `detectorID` of the GuardDuty account in the specific Region where the findings generated.

   To find the `detectorId` for your account and current Region, see the **Settings** page in the https://console.aws.amazon.com/guardduty/ console, or run the ListDetectors API.
5. Replace *`[optional prefix]`* part of the *`S3 bucket ARN/[optional prefix]`* placeholder value with an optional folder location to which you want to export the findings. For more information about the use of prefixes, see Organizing objects using prefixes in the *Amazon S3 User Guide*.

   When you provide an optional folder location that doesn't exist already, GuardDuty will create that location only if the account associated with the S3 bucket is the same as the account exporting the findings. When you export findings to an S3 bucket that belongs to another account, the folder location must exist already.
6. Replace *`KMS key ARN`* with the Amazon Resource Name (ARN) of the KMS key associated with the encryption of the findings exported to the S3 bucket. To locate the key ARN, see Finding the key ID and ARN in the *AWS Key Management Service Developer Guide*.

> **Note**
>
> If you're using GuardDuty in an opt-in Region, replace the value for the "Service" with the Regional endpoint for that Region. For example, if you're using GuardDuty in the Middle East (Bahrain) (me-south-1) Region, replace `"Service": "guardduty.amazonaws.com"` with `"Service": "guardduty.me-south-1.amazonaws.com"`. For information about endpoints for each opt-in Region, see GuardDuty endpoints and quotas.

4. Choose **Save**.

Step 4 - Exporting findings to an S3 bucket (Console)

GuardDuty permits you to export findings to an existing bucket in another AWS account.

When creating a new S3 bucket or choosing an existing bucket in your account, you can add an optional prefix. When configuring export findings, GuardDuty creates a new folder in the S3 bucket for your findings. The prefix will be appended to the default folder structure that GuardDuty created. For example, the format of the optional prefix `/AWSLogs/123456789012/GuardDuty/Region`.

The entire path of the S3 object will be `amzn-s3-demo-bucket/prefix-name/UUID.jsonl.gz`. The `UUID` is randomly generated and doesn't represent the detector ID or the finding ID.

> **Important**
>
> The KMS key and S3 bucket must be in the same Region.

Before completing these steps, make sure you have attached the respective policies to your KMS key and existing S3 bucket.

To configure export findings

1. Open the GuardDuty console at https://console.aws.amazon.com/guardduty/.
2. In the navigation pane, choose **Settings**.
3. On the **Settings** page, under **Findings export options**, for **S3 bucket**, choose **Configure now** (or **Edit**, as needed).
4. For **S3 bucket ARN**, enter the **`bucket ARN`**. To find the bucket ARN, see Viewing the properties for an S3 bucket in the *Amazon S3 User Guide*.
5. For **KMS key ARN**, enter the **`key ARN`**. To locate the key ARN, see Finding the key ID and ARN in the *AWS Key Management Service Developer Guide*.
6. **Attach policies**
   - Perform the steps to attach the S3 bucket policy. For more information, see Step 3 – Attaching policy to Amazon S3 bucket.

- Perform the steps to attach the KMS key policy. For more information, see Step 2 – Attaching policy to your KMS key.

7. Choose **Save**.

Step 5 – Setting frequency to export updated active findings

Configure the frequency for exporting updated active findings as appropriate for your environment. By default, updated findings are exported every 6 hours. This means that any findings that are updated after the most recent export are included in the next export. If updated findings are exported every 6 hours and the export occurs at 12:00, any finding that you update after 12:00 is exported at 18:00.

To set the frequency

1. Open the GuardDuty console at https://console.aws.amazon.com/guardduty/.
2. Choose **Settings**.
3. In the **Findings export options** section, choose **Frequency for updated findings**. This sets the frequency for exporting updated Active findings to both EventBridge and Amazon S3. You can choose from the following:
   - **Update EventBridge and S3 every 15 minutes**
   - **Update EventBridge and S3 every 1 hour**
   - **Update EventBridge and S3 every 6 hours (default)**
4. Choose **Save changes**.

Processing GuardDuty findings with Amazon EventBridge

GuardDuty automatically publishes (sends) findings as events to Amazon EventBridge (formerly Amazon CloudWatch Events), a serverless event bus service. EventBridge delivers a stream of near real-time data from applications and services to targets such as Amazon Simple Notification Service (Amazon SNS) topics, AWS Lambda functions, and Amazon Kinesis streams. For more information, see Amazon EventBridge User Guide.

EventBridge enables automated monitoring and processing of GuardDuty findings by receiving events. EventBridge receives events for both newly generated findings and aggregated findings, where subsequent occurrences of an existing finding are combined with the original. Every GuardDuty finding is assigned a finding ID, and GuardDuty creates an EventBridge event for every finding with a unique finding ID. For information on how aggregation works in GuardDuty, see GuardDuty finding aggregation.

In addition to automated monitoring and processing, use of EventBridge enables longer-term retention of your findings data. GuardDuty stores findings for 90 days. With EventBridge, you can send findings data to your preferred storage platform and store the data for as long as you like. To retain findings for a longer duration, GuardDuty supports Exporting generated findings to Amazon S3.

Topics

- Understanding EventBridge notification frequency in GuardDuty
- Set up an Amazon SNS topic and endpoint (Email, Slack, and Amazon Chime)
- Using Amazon EventBridge for GuardDuty findings
- Creating an EventBridge rule for GuardDuty findings
- EventBridge rule for GuardDuty multi-account environments

Understanding EventBridge notification frequency in GuardDuty

This section explains how often you receive finding notifications through EventBridge and how to update the frequency for subsequent finding occurrences.

Notifications for newly generated findings with a unique finding ID

GuardDuty sends these notifications in near real-time when it generates a finding with a unique finding ID. The notification includes all subsequent occurrences of this subsequent occurrences of this finding ID during the notification generation process.

The notification frequency for newly generated findings is in near real-time. By default, you can not modify this frequency.

Notifications for subsequent finding occurrences

GuardDuty aggregates all subsequent occurrences of a particular finding type that take place within the 6-hour intervals into one single event. Only an administrator account can update the EventBridge notification frequency for subsequent finding occurrences. A member account can't update this frequency for their own account. For example, if the delegated GuardDuty administrator account updates the frequency to one hour, all member accounts will also have one hour notification frequency about the subsequent finding occurrences sent to EventBridge. For more information, see Multiple accounts in Amazon GuardDuty.

As an administrator account, you can customize the default frequency of notifications about the subsequent finding occurrences. Possible values are 15 minutes, 1 hour, or the default 6 hours. For information about setting the frequency for these notifications, see Step 5 – Setting frequency to export updated active findings.

For more details about administrator account receiving EventBridge notifications for member accounts, see EventBridge rule for multi-account environments.

Set up an Amazon SNS topic and endpoint (Email, Slack, and Amazon Chime)

Amazon Simple Notification Service (Amazon SNS) is a fully managed service that provides message delivery from publishers to subscribers. Publishers communicate asynchronously with subscribers by sending messages to a *topic*. A topic is a logical access point and communication channel that lets you group multiple endpoints such as AWS Lambda, Amazon Simple Queue Service (Amazon SQS), HTTP/S, and an email address.

> **Note**
>
> You can add an Amazon SNS topic to your preferred EventBridge event rule during or after the creation of the rule.

Create an Amazon SNS topic

To begin, you must first set up a topic in Amazon SNS and add an endpoint. To create a topic, perform the steps in Step 1: Creating a topic in the *Amazon Simple Notification Service Developer Guide*. After the topic gets created, copy the topic ARN to the clipboard. You will use this topic ARN to continue with one of the preferred setups.

Choose a preferred method to establish where you want to send GuardDuty finding data.

Email setup

To set up an email endpoint

After you Create an Amazon SNS topic, the next step is to create a subscription to this topic. Perform the steps under Step 2: Creating a subscription to an Amazon SNS topic in the *Amazon Simple Notification Service Developer Guide*.

1. For **Topic ARN**, use the topic ARN created in the Create an Amazon SNS topic step. The topic ARN looks similar to the following:

```
arn:aws:sns:us-east-2:123456789012:your_topic
```

2. For **Protocol**, select **Email**.
3. For **Endpoint**, enter an email address where you want to receive the notifications from Amazon SNS.

   After the subscription gets created, you will need to confirm it through your email client.

Slack setup

To configure an Amazon Q Developer in chat applications client - Slack

After you Create an Amazon SNS topic, the next step is to configure the client for Slack.

Perform the steps under Tutorial: Get started with Slack in the *Amazon Q Developer in chat applications Administrator Guide*.

Chime setup

To configure an Amazon Q Developer in chat applications client - Chime

After you Create an Amazon SNS topic, the next step is to configure Amazon Q Developer for Chime.

Perform the steps under Tutorial: Get started with Amazon Chime in the *Amazon Q Developer in chat applications Administrator Guide*.

Using Amazon EventBridge for GuardDuty findings

With EventBridge, you create rules to specify the events that you want to monitor. These rules also specify the target services and applications that can perform automated actions if these events occur. A target is a destination (a resource or an endpoint) that EventBridge sends an event to when the event matches the event pattern defined in the rule. Each event is a JSON object that conforms to the EventBridge schema for AWS events and contains a JSON representation of a finding. You can tailor the rule to send only those events that meet a certain criteria. For more information, see [JSON Schema topic]. Because the findings data is structured as an EventBridge event, you can monitor, process, and act upon findings by using other applications, services, and tools.

In order to receive notifications about GuardDuty findings based on events, you must create an EventBridge rule and a target for GuardDuty. This rule enables EventBridge to send notifications for findings that GuardDuty generates to the target that is specified in the rule.

> **Note**
>
> EventBridge and CloudWatch Events are the same underlying service and API. However, EventBridge includes additional features that help you receive events from software as a service (SaaS) applications and your own applications. Because the underlying service and API are the same, the event schema for GuardDuty findings is also the same.

How archived and non-archived findings in GuardDuty work with EventBridge

For findings that you manually archive, the initial and all subsequent occurrences of these findings (generated after the archiving is complete) are sent to EventBridge based on a specific notification frequency. For more information, see Understanding EventBridge notification frequency in GuardDuty.

For the findings that are automatically archived with Suppression rules, the initial and all subsequent occurrences of these findings (generated after the archiving is complete) are *not* sent to EventBridge. You can view these automatically archived findings in the GuardDuty console.

Event schema

An event pattern defines the data EventBridge uses to determine whether to send the event to the target. The EventBridge event for GuardDuty has the following format:

```
{
         "version": "0",
         "id": "cd2d702e-ab31-411b-9344-793ce56b1bc7",
         "detail-type": "GuardDuty Finding",
         "source": "aws.guardduty",
```

```
        "account": "111122223333",
        "time": "1970-01-01T00:00:00Z",
        "region": "us-east-1",
        "resources": [],
        "detail": {GUARDDUTY_FINDING_JSON_OBJECT}
       }
```

The `detail` value returns the JSON details of a single finding as an object, as opposed to returning the entire *findings* response syntax which supports multiple findings within an array.

For a complete list of all the parameters included in `GUARDDUTY_FINDING_JSON_OBJECT`, see GetFindings. The `id` parameter that appears in `GUARDDUTY_FINDING_JSON_OBJECT` is the finding ID previously described.

Creating an EventBridge rule for GuardDuty findings

The following procedures explain how to use the Amazon EventBridge console and the AWS Command Line Interface (AWS CLI) to create an EventBridge rule for GuardDuty findings. The rule detects EventBridge events that use the event schema and pattern for GuardDuty findings, and it sends those events to an AWS Lambda function for processing.

AWS Lambda is a compute service that you can use to run code without provisioning or managing servers. You package your code and upload it to AWS Lambda as a *Lambda function*. AWS Lambda then runs the function when the function is invoked. A function can be invoked manually by you, automatically in response to events, or in response to requests from applications or services. For information about creating and invoking Lambda functions, see the AWS Lambda Developer Guide.

Choose your preferred method to create an EventBridge rule that sends your GuardDuty finding to a target.

Console

Follow these steps to use the Amazon EventBridge console to create a rule that automatically sends all GuardDuty finding events to a Lambda function for processing. The rule uses default settings for rules that run when specific events are received. For details about rule settings or to learn how to create a rule that uses custom settings, see Creating rules that react to events in the *Amazon EventBridge User Guide*.

Before you create this rule, create the Lambda function that you want the rule to use as a target. When you create the rule, you'll need to specify this function as the target for the rule.

Your target can also be the SNS topic that you created earlier. For more information, see Set up an Amazon SNS topic and endpoint (Email, Slack, and Amazon Chime).

To create an event rule by using console

1. Sign in to the AWS Management Console and open the Amazon EventBridge console at https://console.aws.amazon.com/events/.
2. In the navigation pane, under **Buses**, choose **Rules**.
3. In the **Rules** section, choose **Create rule**.
4. On the **Define rule detail** page, do the following:

   a. For **Name**, enter a name for the rule.

   b. (Optional) For **Description**, enter a brief description of the rule.

   c. For **Event bus**, ensure that **default** is selected and **Enable the rule on the selected event bus** is turned on.

   d. For **Rule type**, choose **Rule with an event pattern**.

   e. When you finish, choose **Next**.
5. On the **Build event pattern** page, do the following:

   a. For **Event source**, choose **AWS events or EventBridge partner events**.

   b. (Optional) For **Sample event**, review a sample finding event for GuardDuty to learn what an event might contain. To do this, choose **AWS events**. Then, for **Sample events**, choose **GuardDuty Finding**.

   c. **Option 1 - Using pattern form, a template that EventBridge provides**

   In the **Event pattern** section, you can do the following:

   1. For **Creation method**, select **Use pattern form**.
   2. For **Event source**, choose **AWS services**.
   3. For **AWS service**, choose **GuardDuty**.
   4. For **Event type**, choose **GuardDuty Finding**.

   When you finish, choose **Next**.

   d. **Option 2 - Using custom event pattern in JSON**

   In the **Event pattern** section, you can do the following:

1. For **Creation method**, select **Custom pattern (JSON editor)**.
2. For **Event pattern**, paste the following custom JSON that will create an alert for medium, high, and critical findings. For more information, see Findings severity levels.

```
{
  "source": [
    "aws.guardduty"
  ],
  "detail-type": [
    "GuardDuty Finding"
  ],
  "detail": {
    "severity": [
      4,
      4.0,
      4.1,
      4.2,
      4.3,
      4.4,
      4.5,
      4.6,
      4.7,
      4.8,
      4.9,
      5,
      5.0,
      5.1,
      5.2,
      5.3,
      5.4,
      5.5,
      5.6,
      5.7,
      5.8,
      5.9,
      6,
      6.0,
      6.1,
      6.2,
      6.3,
      6.4,
```

```
        6.5,
        6.6,
        6.7,
        6.8,
        6.9,
        7,
        7.0,
        7.1,
        7.2,
        7.3,
        7.4,
        7.5,
        7.6,
        7.7,
        7.8,
        7.9,
        8,
        8.0,
        8.1,
        8.2,
        8.3,
        8.4,
        8.5,
        8.6,
        8.7,
        8.8,
        8.9,
        9,
        9.0,
        9.1,
        9.2,
        9.3,
        9.4,
        9.5,
        9.6,
        9.7,
        9.8,
        9.9,
        10,
        10.0
      ]
    }
  }
```

When you finish, choose **Next**.

6. **Option A - Selecting AWS service - AWS Lambda as target**

   On the **Select target(s)** page, do the following:

   a. For **Target types**, select **AWS service**.
   b. For **Select a target**, choose **Lambda function**. Then, for **Function**, choose the Lambda function that you want to send finding events to.
   c. For **Configure version/alias**, enter version or alias settings for the target Lambda function.
   d. (Optional) For **Additional settings**, enter custom settings to specify which event data you want to send to the Lambda function. You can also specify how to handle events that aren't delivered to the function successfully.
   e. When you finish, choose **Next**.

7. **Option B - Selecting SNS topic as target**

   On the **Select target(s)** page, do the following:

   a. For **Target types**, select **AWS service**.
   b. For **Select a target**, choose **SNS topic**. Then, for **Target location**, select the suitable option based on your target location. For **Topic**, choose the name of the SNS topic that you created.
   c. Expand **Additional settings**. For **Configure target input**, choose **Input transformer**.
   d. Choose **Configure input transformer**.
   e. Copy the following code and paste it in the **Input Path** field under the **Target input transformer** section.

```
{
    "severity": "$.detail.severity",
    "Account_ID": "$.detail.accountId",
    "Finding_ID": "$.detail.id",
    "Finding_Type": "$.detail.type",
    "region": "$.region",
    "Finding_description": "$.detail.description"
}
```

f. Copy the following code and paste it into the **Template** field to format the email.

```
"You have a severity <severity> GuardDuty finding type <Finding_Type> in the
 <region> Region."
"Finding Description:"
"<Finding_description>. "
"For more details open the GuardDuty console at https://
console.aws.amazon.com/guardduty/home?region=<region>#/findings?search=id
%3D<Finding_ID>"
```

8. On the **Configure tags** page, optionally enter one or more tags to assign to the rule. Then choose **Next**.

9. On the **Review and create** page, review the rule's settings and verify that they're correct.

   To change a setting, choose **Edit** in the section that contains the setting, and then enter the correct setting. You can also use the navigation tabs to go to the page that contains a setting.

10. When you finish verifying the settings, choose **Create rule**.

API

The following procedure shows how to use AWS CLI commands to create a EventBridge rule and target for GuardDuty. Specifically, the procedure shows you how to create a rule that enables EventBridge to send events for all findings that GuardDuty generates to an AWS Lambda function as a target for the rule.

> **Note**
>
> In this example, we're using a Lambda function as the target for the rule that triggers EventBridge. You can also configure other AWS resources as targets to trigger EventBridge. GuardDuty and EventBridge support the following target types - Amazon EC2 instances, Amazon Kinesis streams, Amazon ECS tasks, AWS Step Functions state machines, the `run` command, and built-in targets. For more information, see [PutTargets](#) in the *Amazon EventBridge API Reference*.

To create a rule and target

1. To create a rule that enables EventBridge to send events for all findings that GuardDuty generates, run the following EventBridge CLI command.

```
aws events put-rule --name your-rule-name --event-pattern "{\"source\":
[\"aws.guardduty\"]}"
```

   You can further customize your rule so that it instructs EventBridge to send events only for a subset of the GuardDuty-generated findings. This subset is based on the finding attribute or attributes that are specified in the rule. For example, use the following CLI command to create a rule that enables EventBridge to only send events for the GuardDuty findings with the severity of either 5 or 8:

```
aws events put-rule --name your-rule-name --event-pattern "{\"source\":
[\"aws.guardduty\"],\"detail-type\":[\"GuardDuty Finding\"],\"detail\":
{\"severity\":[5,8]}}"
```

   For this purpose, you can use any of the property values that are available in the JSON for GuardDuty findings.

2. To attach a Lambda function as a target for the rule that you created in step 1, run the following CloudWatch CLI command.

```
aws events put-targets --rule your-target-name --targets
 Id=1,Arn=arn:aws:lambda:us-east-1:111122223333:function:your_function
```

   Make sure to replace `your-target-name` in the command above with your actual Lambda function for the GuardDuty events.

3. To add the permissions required to invoke the target, run the following Lambda CLI command.

```
aws lambda add-permission --function-name your-target-name --statement-id 1 --
action 'lambda:InvokeFunction' --principal events.amazonaws.com
```

   Make sure to replace `your_function` in the command above with your actual Lambda function for the GuardDuty events.

EventBridge rule for GuardDuty multi-account environments

When using a delegated GuardDuty administrator account, you can view the events generated in the member accounts and take action using other applications and services. EventBridge rules in your administrator account will trigger based on applicable findings from your member accounts. If you set up finding notifications through EventBridge in your administrator account, you will receive notifications of findings from both your account and member accounts. For example, you can use EventBridge to send specific types of findings to a Lambda function that processes and sends the data to your security incident and event management (SIEM) system.

You can identify the member account where the GuardDuty finding originated using the `accountId` field of the finding's JSON details. To create a custom event rule for specific member accounts, create a new rule and use the following template in **Event pattern**. Replace *123456789012* with the `accountId`of the member account for which you want to trigger the event.

```
{
  "source": [
    "aws.guardduty"
  ],
  "detail-type": [
    "GuardDuty Finding"
  ],
  "detail": {
    "accountId": [
      "123456789012"
    ]
  }
}
```

Note

This example creates a rule that matches all findings from the specified account ID. You can include multiple account IDs by separating them with commas, following JSON syntax.

Understanding CloudWatch Logs and reasons for skipping resources during Malware Protection for EC2 scan

GuardDuty Malware Protection for EC2 publishes events to your Amazon CloudWatch log group **/aws/guardduty/malware-scan-events**. For each of the events related to the malware scan, you can monitor the status and scan result of your impacted resources. Certain Amazon EC2 resources and Amazon EBS volumes may have been skipped during the Malware Protection for EC2 scan.

Auditing CloudWatch Logs in GuardDuty Malware Protection for EC2

There are three types of scan events supported in the **/aws/guardduty/malware-scan-events** CloudWatch log group.

Malware Protection for EC2 scan event name	Explanation
`EC2_SCAN_STARTED`	Created when an GuardDuty Malware Protection for EC2 is initiating the process of malware scan, such as preparing to take a snapshot of an EBS volume.
`EC2_SCAN_COMPLETED`	Created when GuardDuty Malware Protection for EC2 scan completes for at least one of the EBS volumes of the impacted resource. This event also includes the `snapshotId` that belongs to the scanned EBS volume. After the scan completes, the scan result will either be `CLEAN`, `THREATS_FOUND` , or `NOT_SCANNED` .
`EC2_SCAN_SKIPPED`	Created when GuardDuty Malware Protectio n for EC2 scan skips all the EBS volumes of the impacted resource. To identify the skip reason, select the corresponding event, and view the details. For more information on skip reasons, see Reasons for skipping resource during malware scan below.

> **Note**
>
> If you're using an AWS Organizations, CloudWatch log events from member accounts in Organizations get published to both administrator account and member account's log group.

Choose your preferred access method to view and query CloudWatch events.

Console

1. Sign in to the AWS Management Console and open the CloudWatch console at https://console.aws.amazon.com/cloudwatch/.
2. In the navigation pane, under **Logs**, choose **Log groups**. Choose the **/aws/guardduty/malware-scan-events** log group to view the scan events for GuardDuty Malware Protection for EC2.

   To run a query, choose **Log Insights**.

   For information about running a query, see Analyzing log data with CloudWatch Logs Insights in the *Amazon CloudWatch User Guide*.
3. Choose **Scan ID** to monitor the details of the impacted resource and malware findings. For example, you can run the following query to filter the CloudWatch log events by using `scanId`. Make sure to use your own valid *`scan-id`*.

```
fields @timestamp, @message, scanRequestDetails.scanId as scanId
| filter scanId like "77a6f6115da4bd95f4e4ca398492bcc0"
| sort @timestamp asc
```

API/CLI

- To work with log groups, see Search log entries using the AWS CLI in the *Amazon CloudWatch User Guide*.

  Choose the **/aws/guardduty/malware-scan-events** log group to view the scan events for GuardDuty Malware Protection for EC2.
- To view and filter log events, see GetLogEvents and FilterLogEvents, respectively, in the *Amazon CloudWatch API Reference*.

GuardDuty Malware Protection for EC2 log retention

The default log retention period for **/aws/guardduty/malware-scan-events** log group is 90 days, after which the log events are deleted automatically. To change the log retention policy for your CloudWatch log group, see [Change log data retention in CloudWatch Logs] in the *Amazon CloudWatch User Guide*, or [PutRetentionPolicy] in the *Amazon CloudWatch API Reference*.

Reasons for skipping resource during malware scan

In the events related to the malware scan, certain EC2 resources and EBS volumes may have been skipped during the scanning process. The following table lists the reasons why GuardDuty Malware Protection for EC2 may not scan the resources. If applicable, use the proposed steps to resolve these issues, and scan these resources the next time GuardDuty Malware Protection for EC2 initiates a malware scan. The other issues are used to inform you about the course of events and are non-actionable.

Reasons for skipping	Explanation	Proposed steps
`RESOURCE_NOT_FOUND`	The `resourceArn` provided to the initiate the on-demand malware scan was not found in your AWS environment.	Validate the `resourceArn` of your Amazon EC2 instance or container workload, and try again.
`ACCOUNT_INELIGIBLE`	The AWS account ID from which you tried initiating an On-demand malware scan has not enabled GuardDuty.	Verify that GuardDuty is enabled for this AWS account. When you enable GuardDuty in a new AWS Region it may take up to 20 minutes to sync.

Reasons for skipping	Explanation	Proposed steps	
`UNSUPPORTED_KEY_ENCRYPTION`	GuardDuty Malware Protection for EC2 supports volumes that are both unencrypted and encrypted with customer managed key. It doesn't support scanning EBS volumes that are encrypted using Amazon EBS encryption. Presently, there is a regional difference where this skip reason is not applicable. For more information about these AWS Regions, see Region-specific feature availability.	Replace your encryption key with a customer managed key. For more information on the types of encryption that GuardDuty supports, see Supported Amazon EBS volumes for malware scan.	

Reasons for skipping	Explanation	Proposed steps
EXCLUDED_BY_SCAN_SETTINGS	The EC2 instance or EBS volume was excluded during the malware scan. There are two possibilities - either the tag was added to the inclusion list but the resource isn't associated with this tag, the tag was added to the exclusion list and the resource is associated with this tag, or the `GuardDutyExcluded` tag is set to `true` for this resource.	Update your scan options or the tags associated to your Amazon EC2 resource. For more information, see Scan options with user-defined tags.
UNSUPPORTED_VOLUME_SIZE	The volume is greater than 2048 GB.	Not actionable.
NO_VOLUMES_ATTACHED	GuardDuty Malware Protection for EC2 found the instance in your account but no EBS volume was attached to this instance to proceed with the scan.	Not actionable.
UNABLE_TO_SCAN	It is an internal service error.	Not actionable.

Reasons for skipping	Explanation	Proposed steps
`SNAPSHOT_NOT_FOUND`	The snapshots created from the EBS volumes and shared with the service account was not found, and GuardDuty Malware Protection for EC2 couldn't proceed with the scan.	Check CloudTrail to ensure that the snapshots were not removed intentionally.
`SNAPSHOT_QUOTA_REACHED`	You have reached the maximum volume allowed for snapshots for each Region. This prevents not just retaining but also creating new snapshots.	You can either remove old snapshots or request for quota increase. You can view the default limit for Snapshots per Region and how to request quota increase under Service quotas in the *AWS General Reference Guide*.
`MAX_NUMBER_OF_ATTACHED_VOLUMES_REACHED`	More than 11 EBS volumes were attached to an EC2 instance. GuardDuty Malware Protection for EC2 scanned the first 11 EBS volumes, obtained by sorting the `deviceName` alphabetically.	Not actionable.

Reasons for skipping	Explanation	Proposed steps	
UNSUPPORTED_PRODUCT_CODE_TYPE	GuardDuty can scan majority of instances with productCode as marketplace . Some marketplace instances may not be eligible for scanning. GuardDuty will skip such instances and log the reason as UNSUPPORTED_PRODUCT_CODE_TYPE . This support varies in AWS GovCloud (US) and China Regions. For more information, see Region-specific feature availability. For more information, see Paid AMIs in the *Amazon EC2 User Guide*. For information on productCode , see ProductCode in the *Amazon EC2 API Reference*.	Not actionable.	

Reporting false positives in Malware Protection for EC2

GuardDuty Malware Protection for EC2 scans may identify a harmless file in your Amazon EC2 instance or container workload as being malicious or harmful. To improve your experience with

Malware Protection for EC2 and the GuardDuty service, you can report false positive results if you believe that a file identified as being malicious or harmful during a scan doesn't actually contain malware.

To report an Amazon EC2 malware scan result as false positive

To initiate the process, contact Support. Use the following steps to provide details about the scanned S3 object:

1. Sign in to the AWS Management Console and open the GuardDuty console at https://console.aws.amazon.com/guardduty/.
2. Choose **EC2 malware Scans**.
3. Choose a scan to view its **Finding ID**.
4. Provide the **Finding ID**. You must also provide the SHA-256 hash of the file. This is required to ensure that GuardDuty Malware Protection for EC2 has received the correct file.
5. The Support team will provide you an Amazon Simple Storage Service (Amazon S3) presigned URL that you can use to upload the potentially malicious file and SHA-256 hash. For information about steps to upload the scanned object, see Uploading objects with presigned URLs in the *Amazon S3 User Guide*.
6. After you have uploaded the file, inform the Support team.

   The Support will provide an acknowledgment after receiving the file. The GuardDuty service team members will analyze your submission, and take appropriate steps to improve your experience with Malware Protection for EC2 and the GuardDuty service. The Support team will continue to provide status update on your case. GuardDuty keeps your S3 object for no more than 30 days.

Reporting S3 object scan result as false positive in Malware Protection for S3

A Malware Protection for S3 scan may identify an object as potentially malicious or harmful. If you believe that the indicated S3 object doesn't contain malware, report this malware scan result as a false positive.

You can submit a false positive report even when you use Malware Protection for S3 independently. In this case, GuardDuty is not designed to generate a finding. For information about checking scan status and result status, see Monitoring S3 object scans.

To report an S3 object malware scan result as false positive

To initiate the process, contact Support. Use the following steps to provide details about the scanned S3 object:

1. Sign in to the AWS Management Console and open the GuardDuty console at https://console.aws.amazon.com/guardduty/.
2. Depending on your use case, choose the appropriate steps:

   Using Malware Protection for S3 with GuardDuty

   1. In the navigation pane, choose **Findings**.
   2. On the **Findings** page, select the false positive finding to view its details.
   3. By checking the finding details, provide the **Finding ID**, **Region**, protected S3 bucket **Name**, and the scanned object **Key**.

      From the **Item path** details, provide the **Hash** of the object. This is required to ensure that GuardDuty has received the correct file.

   Using Malware Protection for S3 independently

   Provide the protected S3 bucket name, scanned object name, and the AWS Region.
3. The Support team will provide you an Amazon Simple Storage Service (Amazon S3) presigned URL that you can use to upload the potentially malicious file and hash. For information about steps to upload the scanned object, see Uploading objects with presigned URLs in the *Amazon S3 User Guide*.
4. After uploading the S3 object, inform the Support team.

The Support will provide an acknowledgment of receiving the object. The GuardDuty service team members will analyze your submission, and take appropriate steps to improve your experience with Malware Protection for S3 and the GuardDuty service. The Support team will continue to provide status update on your case. GuardDuty keeps your S3 object for no more than 30 days.

Remediating detected GuardDuty security findings

Amazon GuardDuty generates findings that indicate potential security findings associated with GuardDuty foundational threat detection and dedicated protection plans. The following sections describe the recommended remediation steps for these scenarios. If there are alternative remediation scenarios, they will be described in the descriptions for each finding type. You can access the full information about a finding type by selecting it from the Active findings types table.

Contents

- Remediating a potentially compromised Amazon EC2 instance
- Remediating a potentially compromised S3 bucket
- Remediating a potentially malicious S3 object
- Remediating a potentially compromised ECS cluster
- Remediating potentially compromised AWS credentials
- Remediating a potentially compromised standalone container
- Remediating EKS Protection findings
- Remediating Runtime Monitoring findings
- Remediating a potentially compromised database
- Remediating a potentially compromised Lambda function

Remediating a potentially compromised Amazon EC2 instance

When GuardDuty generates finding types that indicate potentially compromised Amazon EC2 resources, then your **Resource** will be **Instance**. Potential finding types could be EC2 finding types, GuardDuty Runtime Monitoring finding types, or Malware Protection for EC2 finding types. If the behavior that caused the finding was expected in your environment, then consider using Suppression rules.

Perform the following steps to remediate the potentially compromised Amazon EC2 instance:

1. **Identify the potentially compromised Amazon EC2 instance**

   Investigate the potentially compromised instance for malware and remove any discovered malware. You may use On-demand malware scan in GuardDuty to identify malware in the potentially compromised EC2 instance, or check AWS Marketplace to see if there are helpful partner products to identify and remove malware.

2. **Isolate the potentially compromised Amazon EC2 instance**

   If possible, use the following steps to isolate the potentially compromised instance:

   1. Create a dedicated **Isolation** security group. An isolation security group should only have inbound and outbound access from specific IP addresses. Make sure that there is no inbound or outbound rule that allows traffic for `0.0.0.0/0 (0-65535)`.
   2. Associate the **Isolation** security group with this instance.
   3. Remove all security group associations other than the newly created **Isolation** security group from the potentially compromised instance.

      > **Note**
      >
      > The existing tracked connections won't be terminated as a result of changing security groups - only future traffic will be effectively blocked by the new security group.
      > For information about blocking further traffic from suspicious existing connections, see Enforce NACLs based on network IoCs to prevent further traffic in the *Incident Response Playbook*.

3. **Identify the source of the suspicious activity**

   If malware is detected, then based on the finding type in your account, identify and stop the potentially unauthorized activity on your EC2 instance. This may require actions such as closing any open ports, changing access policies, and upgrading applications to correct vulnerabilities.

   If you are unable to identify and stop unauthorized activity on your potentially compromised EC2 instance, we recommend that you terminate the compromised EC2 instance and replace it with a new instance as needed. The following are additional resources for securing your EC2 instances:

   - Security and Networking sections in Best practices for Amazon EC2
   - Amazon EC2 security groups for Linux instances.
   - Security in Amazon EC2
   - Tips for securing your EC2 instances (Linux).
   - AWS security best practices

- AWS Security Incident Response Technical Guide.

4. **Browse AWS re:Post**

   Browse AWS re:Post for further assistance.

5. **Submit a technical support request**

   If you are a premium support package subscriber, you can submit a technical support request.

Remediating a potentially compromised S3 bucket

When GuardDuty generates GuardDuty S3 Protection finding types, it indicates that your Amazon S3 buckets have been compromised. If the behavior that caused the finding was expected in your environment, then consider creating Suppression rules. If this behavior was not expected, then follow these recommended steps to remediate a potentially compromised Amazon S3 bucket in your AWS environment:

1. **Identify the potentially compromised S3 resource.**

   A GuardDuty finding for S3 will list the associated S3 bucket, its Amazon Resource Name (ARN), and its owner in the finding details.

2. **Identify the source of the suspicious activity and the API call used.**

   The API call used will be listed as `API` in the finding details. The source will be an IAM principal (either an IAM role, user, or account) and identifying details will be listed in the finding. Depending on the source type, Remote IP address or source domain info will be available and can help you evaluate whether the source was authorized. If the finding involved credentials from an Amazon EC2 instance the details for that resource will also be included.

3. **Determine whether the call source was authorized to access the identified resource.**

   For example consider the following:

   - If an IAM user was involved, is it possible that their credentials have been potentially compromised? For more information, see Remediating potentially compromised AWS credentials.
   - If an API was invoked from a principal that has no prior history of invoking this type of API, does this source need access permissions for this operation? Can the bucket permissions be further restricted?

- If the access was seen from the **user name** `ANONYMOUS_PRINCIPAL` with **user type** of `AWSAccount` this indicates the bucket is public and was accessed. Should this bucket be public? If not, review the security recommendations below for alternative solutions to sharing S3 resources.
- If the access was though a successful `PreflightRequest` call seen from the **user name** `ANONYMOUS_PRINCIPAL` with **user type** of `AWSAccount` this indicates the bucket has a cross-origin resource sharing (CORS) policy set. Should this bucket have a CORS policy? If not, ensure the bucket is not inadvertently public and review the security recommendations below for alternative solutions to sharing S3 resources. For more information on CORS see Using cross-origin resource sharing (CORS) in the S3 user guide.

4. **Determine whether the S3 bucket contains sensitive data.**

   Use Amazon Macie to determine whether the S3 bucket contains sensitive data, such as personally identifiable information (PII), financial data, or credentials. If automated sensitive data discovery is enabled for your Macie account, review the S3 bucket's details to gain a better understanding of your S3 bucket's contents. If this feature is disabled for your Macie account, we recommend turning it on to expedite your assessment. Alternatively, you can create and run a sensitive data discovery job to inspect the S3 bucket's objects for sensitive data. For more information, see Discovering sensitive data with Macie.

If the access was authorized, you can ignore the finding. The https://console.aws.amazon.com/guardduty/ console allows you to set up rules to entirely suppress individual findings so that they no longer appear. For more information, see Suppression rules in GuardDuty.

If you determine that your S3 data has been exposed or accessed by an unauthorized party, review the following S3 security recommendations to tighten permissions and restrict access. Appropriate remediation solutions will depend on the needs of your specific environment.

Recommendations based on specific S3 bucket access needs

The following list provides recommendations based on specific Amazon S3 bucket access needs:

- For a centralized way to limit public access to your S3 data use, S3 block public access. Block public access settings can be enabled for access points, buckets, and AWS Accounts through four different settings to control granularity of access. For more information see Block public access settings in the *Amazon S3 User Guide*.

- AWS Access policies can be used to control how IAM users can access your resources or how your buckets can be accessed. For more information see Using bucket policies and user policies in the *Amazon S3 User Guide*.

  Additionally you can use Virtual Private Cloud (VPC) endpoints with S3 bucket policies to restrict access to specific VPC endpoints. For more information see Controlling access from VPC endpoints with bucket policies in the *Amazon S3 User Guide*
- To temporarily allow access to your S3 objects to trusted entities outside your account you can create a Presigned URL through S3. This access is created using your account credentials and depending on the credentials used can last 6 hours to 7 days. For more information see Using presigned URLs to download and upload objects in the *Amazon S3 User Guide*.
- For use cases that require that sharing of S3 objects between different sources you can use S3 Access Points to create permission sets that restrict access to only those within your private network. For more information see Managing access to shared datasets with access points in the *Amazon S3 User Guide*.
- To securely grant access to your S3 resources to other AWS Accounts you can use an access control list (ACL), for more information see Access control list (ACL) overview in the *Amazon S3 User Guide*.

For more information about S3 security options, see Security best practices for Amazon S3 in the *Amazon S3 User Guide*.

Remediating a potentially malicious S3 object

When GuardDuty generates Malware Protection for S3 finding type, it indicates that a newly uploaded object in your Amazon S3 bucket contains malware. The resource type is an **S3Object**.

Use the following recommended steps to potentially remediate the generated finding:

1. Identify the potentially malicious S3 object by checking the **S3ObjectDetails** associated with the finding.
2. Isolate the impacted S3 object. If you had enabled tagging at the time of enabling Malware Protection for S3 for the associated Amazon S3 bucket, GuardDuty must have assigned a **Malicious** tag to this object. Use tag-based access control (TBAC) to restrict access to this S3 object. For more information, see Using tag-based access control (TBAC).

Alternatively, if you do not need this object any longer, you can also choose to delete it or move it to an isolated S3 bucket. For information about considerations for deleting an S3 object, see Deleting objects in the *Amazon S3 User Guide*.

Remediating a potentially compromised ECS cluster

When GuardDuty generates finding types that indicate potentially compromised Amazon ECS resources, then your **Resource** will be **ECSCluster**. Potential finding types could be GuardDuty Runtime Monitoring finding types or Malware Protection for EC2 finding types. If the behavior that caused the finding was expected in your environment, then consider using Suppression rules.

Follow these recommended steps to remediate a potentially compromised Amazon ECS cluster in your AWS environment:

1. **Identify the potentially compromised ECS cluster.**

   The GuardDuty Malware Protection for EC2 finding for ECS provides the **ECS cluster details** in the finding's details panel.
2. **Evaluate the source of malware**

   Evaluate if the detected malware was in the container's image. If malware was in the image, identify all other tasks which are running using this image. For information about running tasks, see ListTasks.
3. **Isolate the potentially impacted tasks**

   Isolate the impacted tasks by denying all ingress and egress traffic to the task. A deny all traffic rule may help you stop an attack that is already underway, by severing all the connections to the task.

If the access was authorized, you can ignore the finding. The https://console.aws.amazon.com/guardduty/ console allows you to set up rules to entirely suppress individual findings so that they no longer appear. For more information, see Suppression rules in GuardDuty.

Remediating potentially compromised AWS credentials

When GuardDuty generates IAM finding types, it indicates that your AWS credentials have been compromised. The potentially compromised **Resource** type is **AccessKey**.

To remediate potentially compromised credentials in your AWS environment, perform the following steps:

1. **Identify the potentially compromised IAM entity and the API call used.**

   The API call used will be listed as `API` in the finding details. The IAM entity (either an IAM role or user) and its identifying information will be listed in the **Resource** section of the finding details. The type of IAM entity involved can be determined by the **User Type** field, the name of the IAM entity will be in the **User name** field. The type of IAM entity involved in the finding can also be determined by the **Access key ID** used.

   For keys beginning with `AKIA`:

      This type of key is a long term customer-managed credential associated with an IAM user or AWS account root user. For information about managing access keys for IAM users, see Managing access keys for IAM users.

   For keys beginning with `ASIA`:

      This type of key is a short term temporary credential generated by AWS Security Token Service. These keys exists for only a short time and cannot be viewed or managed in the AWS Management Console. IAM roles will always use AWS STS credentials, but they can also be generated for IAM Users, for more information on AWS STS see IAM: Temporary security credentials.

      If a role was used the **User name** field will indicate the name of the role used. You can determine how the key was requested with AWS CloudTrail by examining the `sessionIssuer` element of the CloudTrail log entry, for more information see IAM and AWS STS information in CloudTrail.

2. **Review permissions for the IAM entity.**

   Open the IAM console. Depending on the type of the entity used, choose the **Users** or **Roles** tab, and locate the affected entity by typing the identified name into the search field. Use the **Permission** and **Access Advisor** tabs to review effective permissions for that entity.

3. **Determine whether the IAM entity credentials were used legitimately.**

   Contact the user of the credentials to determine if the activity was intentional.

   For example, find out if the user did the following:

   - Invoked the API operation that was listed in the GuardDuty finding

- Invoked the API operation at the time that is listed in the GuardDuty finding
- Invoked the API operation from the IP address that is listed in the GuardDuty finding

If this activity is a legitimate use of the AWS credentials, you can ignore the GuardDuty finding. The https://console.aws.amazon.com/guardduty/ console allows you to set up rules to entirely suppress individual findings so that they no longer appear. For more information, see Suppression rules in GuardDuty.

If you can't confirm if this activity is a legitimate use, it could be the result of a compromise to the particular access key - the IAM user's sign-in credentials, or possibly the entire AWS account. If you suspect your credentials have been compromised, review the information in My AWS account may be compromised to remediate this issue.

Remediating a potentially compromised standalone container

When GuardDuty generates finding types that indicate potentially compromised container, your **Resource type** will be **Container**. If the behavior that caused the finding was expected in your environment, then consider using Suppression rules.

To remediate potentially compromised credentials in your AWS environment, perform the following steps:

1. **Isolate the potentially compromised container**

   The following steps will help you identify the potentially malicious container workload:

   - Open the GuardDuty console at https://console.aws.amazon.com/guardduty/.
   - On the **Findings** page, choose the corresponding finding to view the findings panel.
   - In the findings panel, under the **Resource affected** section, you can view the container's **ID** and **Name**.

   Isolate this container from other container workloads.

2. **Pause the container**

   Suspend all the processes in your container.

   For information about freezing your container, see Pause a container.

   Stop the container.

If the step above fails, and the container doesn't pause, stop the container from running. If you've enabled the Snapshots retention feature, GuardDuty will retain the snapshots of your EBS volumes that contain malware.

For information about stopping the container, see Stop a container.

3. **Evaluate the presence of malware**

   Evaluate if malware was in the container's image.

If the access was authorized, you can ignore the finding. The https://console.aws.amazon.com/guardduty/ console allows you to set up rules to entirely suppress individual findings so that they no longer appear. The GuardDuty console allows you to set up rules to entirely suppress individual findings so that they no longer appear. For more information, see Suppression rules in GuardDuty.

Remediating EKS Protection findings

Amazon GuardDuty generates findings that indicate potential Kubernetes security issues when EKS Protection is enabled for your account. For more information, see EKS Protection. The following sections describe the recommended remediation steps for these scenarios. Specific remediation actions are described in the entry for that specific finding type. You can access the full information about a finding type by selecting it from the Active findings types table.

If any of the EKS Protection finding types were generated expectantly, you can consider adding Suppression rules in GuardDuty to prevent future alerts.

Different types of attacks and configuration issues can trigger GuardDuty EKS Protection findings. This guide helps you identify the root causes of GuardDuty findings against your cluster and outlines appropriate remediation guidance. The following are the primary root causes that lead to GuardDuty Kubernetes findings:

- Potential configuration issues
- Remediating potentially compromised Kubernetes users
- Remediating potentially compromised Kubernetes pods
- Remediating potentially compromised Kubernetes nodes
- Remediating potentially compromised container images

Note

Before Kubernetes version 1.14, the `system:unauthenticated` group was associated to `system:discovery` and `system:basic-user` **ClusterRoles** by default. This may allow unintended access from anonymous users. Cluster updates do not revoke these permissions, which means that even if you have updated your cluster to version 1.14 or later, these permissions may still be in place. We recommend that you disassociate these permissions from the `system:unauthenticated` group.
For more information about removing these permissions, see Secure Amazon EKS clusters with best practices in the Amazon EKS User Guide.

Potential configuration issues

If a finding indicates a configuration issue, see the remediation section of that finding for guidance on resolving that particular issue. For more information, see the following finding types that indicate configuration issues:

- Policy:Kubernetes/AnonymousAccessGranted
- Policy:Kubernetes/ExposedDashboard
- Policy:Kubernetes/AdminAccessToDefaultServiceAccount
- Policy:Kubernetes/KubeflowDashboardExposed
- Any finding that ends in **SuccessfulAnonymousAccess**

Remediating potentially compromised Kubernetes users

A GuardDuty finding can indicate a compromised Kubernetes user when a user identified in the finding has performed an unexpected API action. You can identify the user in the **Kubernetes user details** section of a finding details in the console, or in the `resource.kubernetesDetails.kubernetesUserDetails` of the findings JSON. These user details include `user name`, `uid`, and the Kubernetes groups that the user belongs to.

If the user was accessing the workload using an IAM entity, you can use the `Access Key details` section to identify the details of an IAM role or user. See the following user types and their remediation guidance.

> **Note**
>
> You can use Amazon Detective to further investigate the IAM role or user identified in the finding. While viewing the finding details in GuardDuty console, choose **Investigate in Detective**. Then select AWS user or role from the listed items to investigate it in Detective.

Built-in Kubernetes admin – The default user assigned by Amazon EKS to the IAM identity that created the cluster. This user type is identified by the user name `kubernetes-admin`.

To revoke access of a built-in Kubernetes admin:

- Identify the `userType` from the `Access Key details` section.
  - If the `userType` is **Role** and the role belongs to an EC2 instance role:
    - Identify that instance then follow the instructions in [Remediating a potentially compromised Amazon EC2 instance]().
  - If the `userType` is **User**, or is a **Role** that was assumed by a user:
    1. [Rotate the access key]() of that user.
    2. Rotate any secrets that user had access to.
    3. Review the information in [My AWS account may be compromised]() for further details.

OIDC authenticated user – A user granted access through an OIDC provider. Typically an OIDC user has an email address as a user name. You can check if your cluster uses OIDC with the following command: `aws eks list-identity-provider-configs --cluster-name your-cluster-name`

To revoke access of an OIDC authenticated user:

1. Rotate the credentials of that user in the OIDC provider.
2. Rotate any secrets that user had access to.

AWS-Auth ConfigMap defined user – An IAM user that was granted access through an AWS-auth ConfigMap. For more information, see [Managing users or IAM roles for your cluster]() in the Amazon EKS User Guide. You can review their permissions using the following command: `kubectl edit configmaps aws-auth --namespace kube-system`

To revoke access of an AWS ConfigMap user:

1. Use the following command to open the ConfigMap.

```
kubectl edit configmaps aws-auth --namespace kube-system
```

2. Identify the role or user entry under the **mapRoles** or **mapUsers** section with the same user name as the one reported in the Kubernetes user details section of your GuardDuty finding. See the following example, where the admin user has been identified in a finding.

```
apiVersion: v1
data:
  mapRoles: |
    - rolearn: arn:aws:iam::444455556666:role/eksctl-my-cluster-nodegroup-
standard-wo-NodeInstanceRole-1WP3NUE3O6UCF
      user name: system:node:EC2_PrivateDNSName
      groups:
        - system:bootstrappers
        - system:nodes
  mapUsers: |
    - userarn: arn:aws:iam::123456789012:user/admin
      username: admin
      groups:
        - system:masters
    - userarn: arn:aws:iam::111122223333:user/ops-user
      username: ops-user
      groups:
        - system:masters
```

3. Remove that user from the ConfigMap. See the following example where the admin user has been removed.

```
apiVersion: v1
data:
  mapRoles: |
    - rolearn: arn:aws:iam::111122223333:role/eksctl-my-cluster-nodegroup-
standard-wo-NodeInstanceRole-1WP3NUE3O6UCF
      username: system:node:{{EC2PrivateDNSName}}
      groups:
        - system:bootstrappers
        - system:nodes
  mapUsers: |
    - userarn: arn:aws:iam::111122223333:user/ops-user
      username: ops-user
      groups:
```

```
        - system:masters
```

4. If the `userType` is **User**, or is a **Role** that was assumed by a user:

   a. Rotate the access key of that user.

   b. Rotate any secrets that user had access to.

   c. Review the information in My AWS account may be compromised for further details.

If the finding does not have a `resource.accessKeyDetails` section, the user is a Kubernetes service account.

Service account – The service account provides an identity for pods and can be identified by a user name with the following format: `system:serviceaccount:`*`namespace`*`:`*`service_account_name`*.

To revoke access to a service account:

1. Rotate the service account credentials.
2. Review the guidance for pod compromise in the following section.

Remediating potentially compromised Kubernetes pods

When GuardDuty specifies details of a pod or workload resource inside the `resource.kubernetesDetails.kubernetesWorkloadDetails` section, that pod or workload resource has been potentially compromised. A GuardDuty finding can indicate a single pod has been compromised or that multiple pods have been compromised through a higher-level resource. See the following compromise scenarios for guidance on how to identify the pod or pods that have been compromised.

Single pods compromise

If the `type` field inside the `resource.kubernetesDetails.kubernetesWorkloadDetails` section is **pods**, the finding identifies a single pods. The name field is the `name` of the pods and `namespace` field is its namespace.

For information about identifying the worker node running the pods, see Identify the offending pods and worker node in the *Amazon EKS Best Practices Guide*.

Pods compromised through workload resource

If the `type` field inside the `resource.kubernetesDetails.kubernetesWorkloadDetails` section identifies a **Workload Resource**, such as a `Deployment`, it is likely that all of the pods within that workload resource have been compromised.

For information about identifying all the pods of the workload resource and the nodes on which they are running, see Identify the offending pods and worker nodes using workload name in the *Amazon EKS Best Practices Guide*.

Pods compromised through service account

If a GuardDuty finding identifies a Service Account in the `resource.kubernetesDetails.kubernetesUserDetails` section, it is likely that pods using the identified service account are compromised. The user name reported by a finding is a service account if it has the following format: `system:serviceaccount:namespace:service_account_name`.

For information about identifying all the pods using the service account and the nodes on which they are running, see Identify the offending pods and worker nodes using service account name in the *Amazon EKS Best Practices Guide*.

After you have identified all the compromised pods and the nodes on which they are running, see Isolate the pod by creating a network policy that denies all ingress and egress traffic to the pod in the *Amazon EKS Best Practices Guide*.

To remediate a potentially compromised pod:

1. Identify the vulnerability that compromised the pods.
2. Implement the fix for that vulnerability and start new replacement pods.
3. Delete the vulnerable pods.

   For more information, see Redeploy compromised pod or workload resource in the *Amazon EKS Best Practices Guide*.

If the worker node has been assigned an IAM role that allows Pods to gain access to other AWS resources, remove those roles from the instance to prevent further damage from the attack. Similarly, if the Pod has been assigned an IAM role, evaluate whether you can safely remove the IAM policies from the role without impacting other workloads.

Remediating potentially compromised container images

When a GuardDuty finding indicates a pod compromise, the image used to launch the pod could be potentially malicious or compromised. GuardDuty findings identify the container image within the `resource.kubernetesDetails.kubernetesWorkloadDetails.containers.image` field. You can determine if the image is malicious by scanning it for malware.

To remediate a potentially compromised container image:

1. Stop using the image immediately and remove it from your image repository.
2. Identify all pods using the potentially compromised image.

   For more information, see [Identify pods with vulnerable or compromised images and worker nodes]() in the *Amazon EKS Best Practices Guide*.
3. Isolate the potentially compromised pods, rotate credentials, and gather data for analysis. For more information, see [Isolate the pod by creating a network policy that denies all ingress and egress traffic to the pod]() in the *Amazon EKS Best Practices Guide*.
4. Delete all pods using the potentially compromised image.

Remediating potentially compromised Kubernetes nodes

A GuardDuty finding can indicate a node compromise if the user identified in the finding represents a node identity or if the finding indicates the use of a privileged container.

The user identity is a worker node if the **username** field has following format: `system:node:node name`. For example, `system:node:ip-192-168-3-201.ec2.internal`. This indicates that the adversary has gained access to the node and it is using the node's credentials to talk to the Kubernetes API endpoint.

A finding indicates the use of a privileged container if one or more of the containers listed in the finding has the `resource.kubernetesDetails.kubernetesWorkloadDetails.containers.securityContext.` finding field set to `True`.

To remediate a potentially compromised node:

1. Isolate the pod, rotate its credentials, and gather data for forensic analysis.

For more information, see Isolate the pod by creating a network policy that denies all ingress and egress traffic to the pod in the *Amazon EKS Best Practices Guide*.

2. Identify the service accounts used by all of the pods running on the potentially compromised node. Review their permissions and rotate the service accounts if needed.
3. Terminate the potentially compromised node.

Remediating Runtime Monitoring findings

When you enable Runtime Monitoring for your account, Amazon GuardDuty may generate GuardDuty Runtime Monitoring finding types that indicate potential security issues in your AWS environment. The potential security issues indicate either a compromised Amazon EC2 instance, container workload, an Amazon EKS cluster, or a set of compromised credentials in your AWS environment. The security agent monitors runtime events from multiple resource types. To identify the potentially compromised resource, view **Resource type** in the generated finding details in the GuardDuty console. The following section describes the recommended remediation steps for each resource type.

Instance

If the **Resource type** in the finding details is **Instance**, it indicates that either an EC2 instance or an EKS node is potentially compromised.

- To remediate a compromised EKS node, see Remediating potentially compromised Kubernetes nodes.
- To remediate a compromised EC2 instance, see Remediating a potentially compromised Amazon EC2 instance.

EKSCluster

If the **Resource type** in the finding details is **EKSCluster**, it indicates that either a pod or a container inside an EKS cluster is potentially compromised.

- To remediate a compromised pod, see Remediating potentially compromised Kubernetes pods.
- To remediate a compromised container image, see Remediating potentially compromised container images.

ECSCluster

If the **Resource type** in the finding details is **ECSCluster**, it indicates that either an ECS task or a container inside an ECS task is potentially compromised.

1. **Identify the affected ECS cluster**

   The GuardDuty Runtime Monitoring finding provides the ECS cluster details in the finding's details panel or in the `resource.ecsClusterDetails` section in the finding JSON.
2. **Identify the affected ECS task**

   The GuardDuty Runtime Monitoring finding provides the ECS task details in the finding's details panel or in the `resource.ecsClusterDetails.taskDetails` section in the finding JSON.
3. **Isolate the affected task**

   Isolate the impacted task by denying all ingress and egress traffic to the task. A deny all traffic rule may help stop an attack that is already underway, by severing all connections to the task.
4. **Remediate the compromised task**
   a. Identify the vulnerability that compromised the task.
   b. Implement the fix for that vulnerability and start new a replacement task.
   c. Stop the vulnerable task.

Container

If the **Resource type** in the finding details is **Container**, it indicates that a standalone container is potentially compromised.

- To remediate, see Remediating a potentially compromised standalone container.
- If the finding is generated across multiple containers using the same container image, see Remediating potentially compromised container images.
- If the container has accessed the underlying EC2 host, its associated instance credentials may have been compromised. For more information, see Remediating potentially compromised AWS credentials.
- If a potentially malicious actor has accessed the underlying EKS node or an EC2 instance, see the recommended remediation under the *EKSCluster* and *Instance* tabs.

Remediating compromised container images

When a GuardDuty finding indicates a task compromise, the image used to launch the task could be malicious or compromised. GuardDuty findings identify the container image within the `resource.ecsClusterDetails.taskDetails.containers.image` field. You can determine whether or not the image is malicious by scanning it for malware.

To remediate a compromised container image

1. Stop using the image immediately and remove it from your image repository.
2. Identify all of the tasks that are using this image.
3. Stop all of the tasks that are using the compromised image. Update their task definitions so that they stop using the compromised image.

Remediating a potentially compromised database

GuardDuty generates RDS Protection finding types that indicate potentially suspicious and anomalous login behavior in your Supported databases after you enable RDS Protection. Using RDS login activity, GuardDuty analyzes and profiles threats by identifying unusual patterns in login attempts.

> **Note**
>
> You can access the full information about a finding type by selecting it from the GuardDuty active finding types.

Follow these recommended steps to remediate a potentially compromised Amazon Aurora database in your AWS environment.

Topics

- Remediating potentially compromised database with successful login events
- Remediating potentially compromised database with failed login events
- Remediating potentially compromised credentials
- Restrict network access

Remediating potentially compromised database with successful login events

The following recommended steps can help you remediate a potentially compromised Aurora database that exhibits unusual behavior related to successful login events.

1. **Identify the affected database and user.**

   The generated GuardDuty finding provides the name of the affected database and the corresponding user details. For more information, see Finding details.

2. **Confirm whether this behavior is expected or unexpected.**

   The following list specifies potential scenarios that may have caused GuardDuty to generate a finding:

   - A user who logs in to their database after a long time has passed.
   - A user who logs in to their database on an occasional basis, for example, a financial analyst who logs in each quarter.
   - A potentially suspicious actor who is involved in a successful login attempt potentially compromises the database.

3. **Begin this step if the behavior is unexpected.**

   1. **Restrict database access**

      Restrict database access for the suspected accounts and the source of this login activity. For more information, see Remediating potentially compromised credentials and Restrict network access.

   2. **Assess the impact and determine what information was accessed.**

      - If available, review the audit logs to identify the pieces of information that might have been accessed. For more information, see Monitoring events, logs, and streams in an Amazon Aurora DB cluster in the *Amazon Aurora User Guide*.
      - Determine if any sensitive or protected information was accessed or modified.

Remediating potentially compromised database with failed login events

The following recommended steps can help you remediate a potentially compromised Aurora database that exhibits unusual behavior related to failed login events.

1. **Identify the affected database and user.**

   The generated GuardDuty finding provides the name of the affected database and the corresponding user details. For more information, see Finding details.
2. **Identify the source of the failed login attempts.**

   The generated GuardDuty finding provides the **IP address** and **ASN organization** (if it was a public connection) under the **Actor** section of the finding panel.

   An Autonomous System (AS) is a group of one or more IP prefixes (lists of IP addresses accessible on a network) run by one or more network operators that maintain a single, clearly-defined routing policy. Network operators need Autonomous System Numbers (ASNs) to control routing within their networks and to exchange routing information with other internet service providers (ISPs).
3. **Confirm that this behavior is unexpected.**

   Examine if this activity represents an attempt to gain additional unauthorized access to the database as follows:
   - If the source is internal, examine if an application is misconfigured and attempting a connection repeatedly.
   - If this is an external actor, examine whether the corresponding database is public facing or is misconfigured and thus allowing potential malicious actors to brute force common user names.
4. **Begin this step if the behavior is unexpected.**

   1. **Restrict database access**

      Restrict database access for the suspected accounts and the source of this login activity. For more information, see Remediating potentially compromised credentials and Restrict network access.
   2. **Perform root-cause analysis and determine the steps that potentially led to this activity.**

Set up an alert to get notified when an activity modifies a networking policy and creates an insecure state. For more information, see Firewall policies in AWS Network Firewall in the *AWS Network Firewall Developer Guide*.

Remediating potentially compromised credentials

A GuardDuty finding may indicate that the user credentials for an affected database have been compromised when the user identified in the finding has performed an unexpected database operation. You can identify the user in the **RDS DB user details** section within the finding panel in the console, or within the `resource.rdsDbUserDetails` of the findings JSON. These user details include user name, application used, database accessed, SSL version, and authentication method.

- To revoke access or rotate passwords for specific users that are involved in the finding, see Security with Amazon Aurora MySQL, or Security with Amazon Aurora PostgreSQL in the *Amazon Aurora User Guide*.
- Use AWS Secrets Manager to securely store and automatically rotate the secrets for Amazon Relational Database Service(RDS) databases. For more information, see AWS Secrets Manager tutorials in the *AWS Secrets Manager User Guide*.
- Use IAM database authentication to manage database users' access without the need for passwords. For more information, see IAM database authentication in the *Amazon Aurora User Guide*.

  For more information, see Security best practices for Amazon Relational Database Service in the *Amazon RDS User Guide*.

Restrict network access

A GuardDuty finding may indicate that a database is accessible beyond your applications, or Virtual Private Cloud (VPC). If the remote IP address in the finding is an unexpected connection source, audit the security groups. A list of security groups attached to the database is available under **Security groups** in the https://console.aws.amazon.com/rds/ console, or in the `resource.rdsDbInstanceDetails.dbSecurityGroups` of the findings JSON. For more information on configuring security groups, see Controlling access with security groups in the *Amazon RDS User Guide*.

If you're using a firewall, restrict network access to the database by reconfiguring the Network Access Control Lists (NACLs). For more information, see Firewalls in AWS Network Firewall in the *AWS Network Firewall Developer Guide*.

Remediating a potentially compromised Lambda function

When GuardDuty generates Lambda Protection finding types, your Lambda function may be compromised. If the activity that caused GuardDuty to generate this finding was expected, you can consider using Suppression rules. We recommend completing the following steps to remediate a compromised Lambda function:

To remediate Lambda Protection findings

1. **Identify the potentially compromised Lambda function version**.

   A GuardDuty finding for Lambda Protection provides the name, Amazon Resource Name (ARN), function version, and revision ID associated with the Lambda function listed in the finding details.
2. **Identify the source of the potentially suspicious activity**.
   a. Review the code associated with the Lambda function version involved in the finding.
   b. Review the imported libraries and layers of the Lambda function version involved in the finding.
   c. If you have enabled Scanning AWS Lambda functions with Amazon Inspector, review the Amazon Inspector findings associated with the Lambda function involved in the finding.
   d. Review the AWS CloudTrail logs to identify the principal that caused the function update and ensure that the activity was authorized or expected.
3. **Remediate the potentially compromised Lambda function**.
   a. Disable the execution triggers of the Lambda function involved in the finding. For more information, see DeleteFunctionEventInvokeConfig.
   b. Review the Lambda code and update the libraries imports and Lambda function layers to remove the potentially suspicious libraries and layers.
   c. Mitigate Amazon Inspector findings related to the Lambda function involved in the finding.

Estimating GuardDuty usage cost

During the 30-day free trial, you can use the GuardDuty console or API operations to estimate the daily average usage costs for GuardDuty. The cost estimation projects what your estimated costs will be after the trial period. However, to review an accurate cost estimate during free trial, GuardDuty recommends using AWS Billing at https://console.aws.amazon.com/costmanagement/.

When you operate in a multiple-account environment, the GuardDuty administrator account can monitor cost metrics for all of the member accounts.

> **Note about Malware Protection for S3 usage cost**
>
> The usage cost for Malware Protection for S3 is not included under **Usage** in the GuardDuty console. For more information, see Reviewing usage cost for Malware Protection for S3.

You can view cost estimation based on the following metrics:

- **Account ID** – Lists the estimated cost for your account, or for your member accounts if you are operating as a GuardDuty administrator account.
- **Data sources** – Lists the estimated cost for all the Foundational data sources – AWS CloudTrail management events, VPC flow logs, and Route53 Resolver DNS query logs.
- **Features** – Lists the estimated cost for the GuardDuty features – CloudTrail data events for S3, EKS Audit Log Monitoring, EBS volume data, RDS login activity, EKS Runtime Monitoring, Fargate Runtime Monitoring, EC2 Runtime Monitoring, or Lambda Network Activity Monitoring.
- **S3 buckets** – Lists the estimated cost for S3 data events on a specified bucket or the most expensive buckets for accounts in your environment. This statistic is available only when you enable S3 Protection for an AWS account.

Understanding how GuardDuty calculates usage costs

The estimates displayed in the GuardDuty console may differ slightly than those in your AWS Billing and Cost Management console. The following list explains how GuardDuty estimates usage costs:

- The GuardDuty usage estimate is for the current Region only.

- The GuardDuty usage cost is based on the last 30 days of usage.
- The trial usage cost estimate includes the estimate for foundational data sources and features that are currently in the trial period. Each feature and data source within GuardDuty has its own trial period but it may overlap with the trial period of GuardDuty or another feature that was enabled at the same time.
- The GuardDuty usage estimate includes GuardDuty volume pricing discounts per Region, as detailed on the Amazon GuardDuty Pricing page, but only for individual accounts meeting the volume pricing tiers. Volume pricing discounts are not included in estimates for combined total usage between accounts within an organization. For information about combined usage volume discount pricing, see AWS Billing: Volume Discounts.
- The sum of the usage cost for each AWS account in your organization may not always be the same as the last 30-day estimated cost for the selected data source. The pricing tier may change as GuardDuty processes more events or data. For more information, see Pricing Tiers in the *AWS Billing User Guide*.

This scenario explains that to stop incurring usage cost for Runtime Monitoring, you must have both the Runtime Monitoring and EKS Runtime Monitoring features disabled.

GuardDuty has consolidated the console experience for EKS Runtime Monitoring into Runtime Monitoring. GuardDuty recommends Checking EKS Runtime Monitoring configuration status and Migrating from EKS Runtime Monitoring to Runtime Monitoring.

As a part of migrating to Runtime Monitoring, ensure to Disable EKS Runtime Monitoring. This is important because if you later choose to disable Runtime Monitoring and you do not disable EKS Runtime Monitoring, you will continue incurring usage cost for EKS Runtime Monitoring.

Runtime Monitoring – How VPC flow logs from EC2 instances impact usage cost

When you manage the security agent (either manually or through GuardDuty) in EKS Runtime Monitoring or Runtime Monitoring for EC2 instances, and GuardDuty is presently deployed on an Amazon EC2 instance and receives the Collected runtime event types from this instance, GuardDuty will not charge your AWS account for the analysis of VPC flow logs from this Amazon EC2 instance. This helps GuardDuty avoid double usage cost in the account.

How GuardDuty estimates usage cost for CloudTrail events

When you enable GuardDuty, it automatically starts consuming AWS CloudTrail event logs recorded for your account in the selected AWS Region. GuardDuty replicates Global service events logs and then processes these events independently in each Region where you have GuardDuty enabled. This helps GuardDuty maintain user and role profiles in each Region to identify anomalies.

Your CloudTrail configuration does not impact GuardDuty usage cost or the way GuardDuty processes your event logs. Your GuardDuty usage cost is affected by your usage of AWS APIs which log to CloudTrail. For more information, see AWS CloudTrail management events.

Reviewing GuardDuty estimated usage cost

The GuardDuty usage provides cost estimates based on the your usage over the last 30 days per AWS Region. The estimated usage is different than your billing usage. For information about how GuardDuty estimates the usage cost, see Understanding how GuardDuty calculates usage costs. If you're a GuardDuty administrator account, you can view the cost estimates for each member account, broken down by data sources and accounts.

Choose your preferred access method to review the usage cost for your GuardDuty account.

To review estimated GuardDuty usage cost

Console

1. Open the GuardDuty console at https://console.aws.amazon.com/guardduty/.

   Make sure to use the GuardDuty administrator account.
2. In the navigation pane, choose **Usage**.
3. On the **Usage** page, a GuardDuty administrator account with member accounts can view the **Estimated organization cost** for the last 30 days. This is an estimated total usage cost for your organization.
4. GuardDuty administrator accounts can either view the usage cost breakdown by data source, or by accounts. Individual or standalone accounts can view the breakdown by data source.

   If you have member accounts – Select the **By accounts** tab to view the statistics for each member account.

Under the **By data sources** tab, when you select a data source that has a usage cost associated with it, the corresponding sum of the cost breakdown at the accounts level may not always be the same.

API/CLI

Run the GetUsageStatistics API operation using the credentials of GuardDuty administrator account account. Provide the following information to run the command:

- (Required) provide the Regional GuardDuty detector ID of the account for which you want to retrieve the statistics.
- (Required) provide one of the types of statistics to retrieve: `SUM_BY_ACCOUNT | SUM_BY_DATA_SOURCE | SUM_BY_RESOURCE | SUM_BY_FEATURE | TOP_ACCOUNTS_BY_FEATURE`.

  Currently, `TOP_ACCOUNTS_BY_FEATURE` does not support retrieving usage statistics for `RDS_LOGIN_EVENTS`.
- (Required) provide one or more data sources or features to query your usage statistics.
- (Optional) provide a list of account IDs for which you want to retrieve usage statistics.

You can also use the AWS Command Line Interface. The following command is an example about retrieving the usage statistics for all the data sources and features, calculated by accounts. Make sure to replace the `detector-id` with your own valid detector ID. For standalone accounts, this command returns the usage cost over the past 30 days for your account only. If you are a GuardDuty administrator account with member accounts, you see costs listed by account for all members.

To find the `detectorId` for your account and current Region, see the **Settings** page in the https://console.aws.amazon.com/guardduty/ console, or run the ListDetectors API.

Replace `SUM_BY_ACCOUNT` by the type with which you want to calculate the usage statistics.

To monitor cost for data sources only

```
aws guardduty get-usage-statistics --detector-id 12abc34d567e8fa901bc2d34e56789f0
 --usage-statistic-type SUM_BY_ACCOUNT --usage-criteria '{"DataSources":
```

```
["FLOW_LOGS", "CLOUD_TRAIL", "DNS_LOGS", "S3_LOGS", "KUBERNETES_AUDIT_LOGS",
 "EC2_MALWARE_SCAN"]}'
```

To monitor cost for features

```
aws guardduty get-usage-statistics --detector-id 12abc34d567e8fa901bc2d34e56789f0
 --usage-statistic-type SUM_BY_ACCOUNT --usage-criteria '{"Features":
["FLOW_LOGS", "CLOUD_TRAIL", "DNS_LOGS", "S3_DATA_EVENTS", "EKS_AUDIT_LOGS",
 "EBS_MALWARE_PROTECTION", "RDS_LOGIN_EVENTS", "LAMBDA_NETWORK_LOGS",
 "EKS_RUNTIME_MONITORING", "FARGATE_RUNTIME_MONITORING", "EC2_RUNTIME_MONITORING"]}'
```

Feature names for protection plans in GuardDuty API

When you enable Amazon GuardDuty for the first time, it starts processing Foundational data sources within your AWS environment. GuardDuty uses these data sources to process an independent stream of events such as VPC flow logs, DNS logs, and AWS CloudTrail management events. It then analyzes these events to identify potential security threats and generates findings in your account.

When one or more protection plans are enabled, then GuardDuty uses additional data from other AWS services in your AWS environment to monitor and analyze for potential security threats. These additional data sources are called features.

Change from data sources to features

When you add additional GuardDuty protections, such as S3 Protection, Runtime Monitoring, Lambda Protection, and others, you can configure the GuardDuty feature corresponding to the protection plan. Historically, GuardDuty protections were called `dataSources` in the APIs. However, after March 2023, new GuardDuty protection plans are now configured as `features` and not `dataSources`. GuardDuty still supports configuring protection plans launched before March 2023, as `dataSources` through the API, but new protection plans are only available as `features`. For information about which protection plans are impacted, see GuardDuty API changes.

If you manage GuardDuty configuration and protection plans through the console, you are not directly impacted by this change and don't need to take any action. This change affects the behavior of the APIs that are invoked to enable GuardDuty or protection plans within GuardDuty. If you use APIs or AWS CLI to enable or edit the configuration of a protection plan, you must use the associated feature name. For more information, see Mapping `dataSources` to `features`.

GuardDuty API changes in March 2023

The GuardDuty APIs configure protection features that don't belong to the list of GuardDuty foundational data sources. A feature object contains feature details, such as feature name and status, and may contain additional configuration for some of the protection plans. This migration affects the following APIs in the *Amazon GuardDuty API Reference*:

- CreateDetector
- GetDetector

- UpdateDetector
- GetMemberDetectors
- UpdateMemberDetectors
- DescribeOrganizationConfiguration
- UpdateOrganizationConfiguration
- GetRemainingFreeTrialDays
- GetUsageStatistics

Features compared to data sources

Historically, all GuardDuty features were passed through a `dataSources` object in the API. From March 2023, GuardDuty prefers `features` object instead of the `dataSources` object in the API. All earlier data sources have corresponding features, but newer features may not have corresponding data sources.

The following list shows the comparison between `dataSources` and `features` object when passed through an API:

- The `dataSources` object contains objects for each protection type and its status. The `features` object is a list of available features that correspond to each protection type within GuardDuty.

  Starting March 2023, feature activation will be the only way to configure new GuardDuty features in your AWS environment.
- The `dataSources` schema in the API request or response is the same in each AWS Region where GuardDuty is available. However, every feature may not be available in each Region. Therefore, the available feature names may differ based on the Region.

Understanding how APIs with features work

The GuardDuty APIs will continue to return a `dataSources` object as applicable, and they will also return a `features` object containing the same information in a different format. GuardDuty features launched before March 2023 will be available through `dataSources` object and `features` object. GuardDuty launched features since March 2023 will only be available through the `features` object. You can't create or update a detector, or describe your AWS Organizations using both `dataSources` and `features` object notation in the same API request. To enable

GuardDuty protection types, you will need to migrate your existing data sources to the `features` object by using the same APIs that now include the `features` object too.

Note

GuardDuty will not add new data source after this modification.

GuardDuty has deprecated the use of data sources that are associated with the protection plans. However, it still supports the GuardDuty foundational data sources. The GuardDuty best practices recommend using features for enabling or editing the configuration for any protection plan in your account.

Incorporating feature changes in APIs

- If you manage GuardDuty configurations through APIs, SDKs, or AWS CloudFormation template, and want to enable potential new GuardDuty features, you will need to modify your code and template, respectively. For more information, see the updated APIs in the *Amazon GuardDuty API Reference*.
- For GuardDuty features configured prior to this upgrade, you can continue using the APIs, SDKs, or AWS CloudFormation template. However, we recommend that you switch to using `feature` object.

  All the data sources have an equivalent feature object. For more information, see Mapping `dataSources` to `features`.
- Presently, `additionalConfiguration` in the `features` object is only available for certain protection types.
  - For such protection types, if your feature's `AdditionalConfiguration` `status` is set to `ENABLED` but your feature's configuration `status` is not set to `ENABLED`, GuardDuty will not take any action in this case.
  - The following APIs get impacted by this:
    - UpdateDetector
    - UpdateMemberDetectors
    - UpdateOrganizationConfiguration

Mapping `dataSources` to `features`

The following table shows the mapping of protection types, `dataSources`, and `features`.

GuardDuty protection type	Data source name*	Feature name
VPC Flow Logs	`flowLogs` (read only; can't be modified)	`FLOW_LOGS` (read only; can't be modified)
Route53 Resolver DNS query logs	`dnsLogs` (read only; can't be modified)	`DNS_LOGS` (read only; can't be modified)
CloudTrail events	`cloudTrail` (read only; can't be modified)	`CLOUD_TRAIL` (read only; can't be modified)
S3	`s3Logs`	`S3_DATA_EVENTS`
EKS Protection	`kubernetes.auditlogs`	`EKS_AUDIT_LOGS`
Malware Protection for EC2	`malwareProtection.scanEc2InstanceWithFindings.ebsVolumes`	`EBS_MALWARE_PROTECTION`
RDS Login events	GuardDuty provides only feature activation support for these protection types.	`RDS_LOGIN_EVENTS`
EKS Runtime Monitoring		`EKS_RUNTIME_MONITORING`
Runtime Monitoring		`RUNTIME_MONITORING`
GuardDuty security agent for Amazon EKS clusters		`EKS_RUNTIME_MONITORING.additionalCon`

GuardDuty protection type	Data source name*	Feature name
		figuration.EKS_ADDON_MANAGEMENT RUNTIME_MONITORING.additionalConfiguration.EKS_ADDON_MANAGEMENT
GuardDuty security agent for Amazon ECS-Fargate clusters		RUNTIME_MONITORING.additionalConfiguration.ECS_FARGATE_AGENT_MANAGEMENT
GuardDuty security agent for Amazon EC2 instances		RUNTIME_MONITORING.additionalConfiguration.EC2_AGENT_MANAGEMENT
Lambda Protection		LAMBDA_NETWORK_LOGS

*GetUsageStatistics uses its own `dataSource` names. For more information, see Estimating GuardDuty usage cost or GetUsageStatistics.

Security in Amazon GuardDuty

Cloud security at AWS is the highest priority. As an AWS customer, you benefit from a data center and network architecture that is built to meet the requirements of the most security-sensitive organizations.

Security is a shared responsibility between AWS and you. The Shared Responsibility Model describes this as security of the cloud and security in the cloud:

- **Security of the cloud** – AWS is responsible for protecting the infrastructure that runs AWS services in the AWS Cloud. AWS also provides you with services that you can use securely. Third-party auditors regularly test and verify the effectiveness of our security as part of the AWS compliance programs. To learn about the compliance programs that apply to GuardDuty, see AWS services in scope by compliance program.
- **Security in the cloud** – Your responsibility is determined by the AWS service that you use. You are also responsible for other factors including the sensitivity of your data, your company's requirements, and applicable laws and regulations.

This documentation helps you understand how to apply the shared responsibility model when using GuardDuty. It shows you how to configure GuardDuty to meet your security and compliance objectives. You also learn how to use other AWS services that help you to monitor and secure your GuardDuty resources.

Contents

- Data protection in Amazon GuardDuty
- Logging Amazon GuardDuty API calls with AWS CloudTrail
- Identity and Access Management for Amazon GuardDuty
- Compliance validation for Amazon GuardDuty
- Resilience in Amazon GuardDuty
- Infrastructure security in Amazon GuardDuty
- Amazon GuardDuty and interface VPC endpoints (AWS PrivateLink)

Data protection in Amazon GuardDuty

The AWS shared responsibility model applies to data protection in Amazon GuardDuty. As described in this model, AWS is responsible for protecting the global infrastructure that runs all of the AWS Cloud. You are responsible for maintaining control over your content that is hosted on this infrastructure. You are also responsible for the security configuration and management tasks for the AWS services that you use. For more information about data privacy, see the Data Privacy FAQ. For information about data protection in Europe, see the AWS Shared Responsibility Model and GDPR blog post on the *AWS Security Blog*.

For data protection purposes, we recommend that you protect AWS account credentials and set up individual users with AWS IAM Identity Center or AWS Identity and Access Management (IAM). That way, each user is given only the permissions necessary to fulfill their job duties. We also recommend that you secure your data in the following ways:

- Use multi-factor authentication (MFA) with each account.
- Use SSL/TLS to communicate with AWS resources. We require TLS 1.2 and recommend TLS 1.3.
- Set up API and user activity logging with AWS CloudTrail. For information about using CloudTrail trails to capture AWS activities, see Working with CloudTrail trails in the *AWS CloudTrail User Guide*.
- Use AWS encryption solutions, along with all default security controls within AWS services.
- Use advanced managed security services such as Amazon Macie, which assists in discovering and securing sensitive data that is stored in Amazon S3.
- If you require FIPS 140-3 validated cryptographic modules when accessing AWS through a command line interface or an API, use a FIPS endpoint. For more information about the available FIPS endpoints, see Federal Information Processing Standard (FIPS) 140-3.

We strongly recommend that you never put confidential or sensitive information, such as your customers' email addresses, into tags or free-form text fields such as a **Name** field. This includes when you work with GuardDuty or other AWS services using the console, API, AWS CLI, or AWS SDKs. Any data that you enter into tags or free-form text fields used for names may be used for billing or diagnostic logs. If you provide a URL to an external server, we strongly recommend that you do not include credentials information in the URL to validate your request to that server.

Encryption at rest

All GuardDuty customer data is encrypted at rest using AWS encryption solutions.

GuardDuty data, such as findings, is encrypted at rest using AWS Key Management Service (AWS KMS) using AWS owned customer managed keys.

Encryption in transit

GuardDuty analyzes log data from other services. It encrypts all data in transit from these services with HTTPS and KMS. Once GuardDuty extracts the information it needs from the logs, they are discarded. For more information on how GuardDuty uses information from other services, see GuardDuty data sources.

GuardDuty data is encrypted in transit between services.

Opting out of using your data for service improvement

You can choose to opt out of having your data used to develop and improve GuardDuty and other AWS security services by using the AWS Organizations opt-out policy. You can choose to opt out even if GuardDuty doesn't currently collect any such data. For more information about how to opt out, see AI services opt-out policies in the *AWS Organizations User Guide*.

> **Note**
>
> For you to use the opt-out policy, your AWS accounts must be centrally managed by AWS Organizations. If you haven't already created an organization for your AWS accounts, see Creating and managing an organization in the *AWS Organizations User Guide*.

Opting out has the following effects:

- GuardDuty will delete the data that it collected and stored for service improvement purposes prior to your opt out (if any).
- After you opt out, GuardDuty will no longer collect or store this data for service improvement purposes.

The following topics explain how each feature within GuardDuty potentially handles your data for service improvement.

Contents

- GuardDuty Runtime Monitoring

- GuardDuty Malware Protection

GuardDuty Runtime Monitoring

GuardDuty Runtime Monitoring provides runtime threat detection for Amazon Elastic Kubernetes Service (Amazon EKS) clusters, AWS Fargate Amazon Elastic Container Service(Amazon ECS) only, and Amazon Elastic Compute Cloud (Amazon EC2) instances in your AWS environment. After you enable Runtime Monitoring and deploy the GuardDuty security agent for your resource, GuardDuty starts to monitor and analyze the runtime events associated with your resource. These runtime event types include process events, container events, DNS events, and more. For more information, see Collected runtime event types that GuardDuty uses.

GuardDuty collects both commands (such as `curl`, `systemctl`, and `cron`) and their associated arguments (such as `start`, `stop`, `disable`) from your workloads. For example, when someone runs `systemctl stop service-name`, GuardDuty captures both the command `systemctl` and its arguments `stop service-name`. This detailed information helps GuardDuty to detect sophisticated threats by analyzing command patterns and correlating multiple events. For example, GuardDuty can identify when an attacker attempts to disable security services or executes known malicious files. While GuardDuty actively uses this data for threat detection, it **doesn't** currently use these commands and arguments for service improvement purposes (it may do so in the future). Your trust, privacy, and the security of your content are our highest priority, and ensure that our use complies with our commitments to you. For more information, see Data Privacy FAQ.

GuardDuty Malware Protection

GuardDuty Malware Protection scans and detects malware contained in EBS volumes attached to your potentially compromised Amazon EC2 instance and container workloads, and newly uploaded files in your selected Amazon S3 buckets. Currently, GuardDuty doesn't collect or use detected malware for service improvement. However, in the future, when GuardDuty Malware Protection identifies an EBS volume file or an S3 file as being malicious or harmful, GuardDuty Malware Protection will collect and store this file to develop and improve its malware detections, and the GuardDuty service. This file may also be used to develop and improve other AWS security services. Your trust, privacy, and the security of your content are our highest priority, and ensure that our use complies with our commitments to you. For more information, see Data Privacy FAQ.

Logging Amazon GuardDuty API calls with AWS CloudTrail

Amazon GuardDuty is integrated with AWS CloudTrail, a service that provides a record of actions taken by a user, role, or an AWS service in GuardDuty. CloudTrail captures all API calls for GuardDuty as events, including calls from the GuardDuty console and from code calls to the GuardDuty APIs. If you create a trail, you can enable continuous delivery of CloudTrail events to an Amazon Simple Storage Service (Amazon S3) bucket, including events for GuardDuty. If you don't configure a trail, you can still view the most recent events in the CloudTrail console in **Event history**. Using the information collected by CloudTrail, you can determine the request that was made to GuardDuty, the IP address the request was made from, who made the request, when it was made, and additional details.

For more information about CloudTrail, including how to configure and enable it, see the *AWS CloudTrail User Guide*.

GuardDuty information in CloudTrail

CloudTrail is enabled on your AWS account when you create the account. When supported event activity occurs in GuardDuty, that activity is recorded in a CloudTrail event along with other AWS service events in **Event history**. You can view, search, and download recent events in your AWS account. For more information, see Viewing events with CloudTrail event history.

For an ongoing record of events in your AWS account, including events for GuardDuty, create a trail. A trail enables CloudTrail to deliver log files to an Amazon S3 bucket. By default, when you create a trail in the console, the trail applies to all Regions. The trail logs events from all Regions in the AWS partition and delivers the log files to the Amazon S3 bucket that you specify. Additionally, you can configure other AWS services to further analyze and act upon the event data collected in CloudTrail logs. For more information, see:

- Overview for creating a trail
- CloudTrail supported services and integrations
- Configuring Amazon SNS notifications for CloudTrail
- Receiving CloudTrail log files from multiple regions and Receiving CloudTrail log files from multiple accounts

Every event or log entry contains information about who generated the request. The identity information helps you determine the following:

- Whether the request was made with root user or IAM user's sign-in credentials
- Whether the request was made with temporary security credentials for a role or federated user
- Whether the request was made by another AWS service

For more information, see CloudTrail userIdentity element.

GuardDuty control plane events in CloudTrail

By default, CloudTrail logs all the GuardDuty API operations provided in the Amazon GuardDuty API Reference as events in CloudTrail files.

GuardDuty data events in CloudTrail

GuardDuty Runtime Monitoring uses a GuardDuty security agent deployed to your Amazon Elastic Kubernetes Service (Amazon EKS) clusters, Amazon Elastic Compute Cloud (Amazon EC2) instances, and AWS Fargate (Amazon Elastic Container Service (Amazon ECS) only) tasks to collect add-on (`aws-guardduty-agent`) that collects Collected runtime event types for your AWS workloads and then send them to GuardDuty for threat detection and analysis.

Logging and monitoring data events

You can optionally configure the AWS CloudTrail logs to view the data events for your GuardDuty security agent.

To create and configure CloudTrail, see Data events in the *AWS CloudTrail User Guide* and follow the instructions for **Logging data events with advanced event selectors in the AWS Management Console**. While logging the trail, ensure to make the following changes:

- For the **Data event type**, choose **GuardDuty detector**.
- For the **Log selector template**, choose **Log all events**.
- Expand the **JSON view** for the configuration. It should be similar to the following JSON:

```
[
  {
    "name": "",
    "fieldSelectors": [
      {
        "field": "eventCategory",
        "equals": [
```

```
          "Data"
        ]
      },
      {
        "field": "resources.type",
        "equals": [
          "AWS::GuardDuty::Detector"
        ]
      }
    ]
  }
]
```

After you enable the selector for the trail, navigate to the Amazon S3 console at https://console.aws.amazon.com/s3/. You can download the data events from your S3 bucket chosen at the time of configuring the CloudTrail logs.

Example: GuardDuty log file entries

A trail is a configuration that enables delivery of events as log files to an Amazon S3 bucket that you specify. CloudTrail log files contain one or more log entries. An event represents a single request from any source and includes information about the requested action, the date and time of the action, request parameters, and so on. CloudTrail log files aren't an ordered stack trace of the public API calls, so they don't appear in any specific order.

The following example shows a CloudTrail log entry that demonstrates the data plane event.

```
{
        "eventVersion": "1.08",
        "userIdentity": {
            "type": "AssumedRole",
            "principalId": "111122223333:aws:ec2-instance:i-123412341234example",
            "arn": "arn:aws:sts::111122223333:assumed-role/aws:ec2-
instance/i-123412341234example",
            "accountId": "111122223333",
            "accessKeyId": "AKIAI44QH8DHBEXAMPLE",
            "sessionContext": {
                "sessionIssuer": {
                    "type": "Role",
                    "principalId": "111122223333:aws:ec2-instance",
                    "arn": "arn:aws:iam::111122223333:role/aws:ec2-instance",
```

```
                    "accountId": "111122223333",
                    "userName": "aws:ec2-instance"
                },
                "attributes": {
                    "creationDate": "2023-03-05T04:00:21Z",
                    "mfaAuthenticated": "false"
                },
                "ec2RoleDelivery": "2.0"
            }
        },
        "eventTime": "2023-03-05T06:03:49Z",
        "eventSource": "guardduty.amazonaws.com",
        "eventName": "SendSecurityTelemetry",
        "awsRegion": "us-east-1",
        "sourceIPAddress": "54.240.230.177",
        "userAgent": "aws-sdk-rust/0.54.1 os/linux lang/rust/1.66.0",
        "requestParameters": null,
        "responseElements": null,
        "requestID": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
        "eventID": "a1b2c3d4-5678-90ab-cdef-EXAMPLEbbbbb",
        "readOnly": false,
        "resources": [{
            "accountId": "111122223333",
            "type": "AWS::GuardDuty::Detector",
            "ARN": "arn:aws:guardduty:us-
west-2:111122223333:detector/12abc34d567e8fa901bc2d34e56789f0"
        }],
        "eventType": "AwsApiCall",
        "managementEvent": false,
        "recipientAccountId": "111122223333",
        "eventCategory": "Data",
        "tlsDetails": {
            "tlsVersion": "TLSv1.2",
            "cipherSuite": "ECDHE-RSA-AES128-GCM-SHA256",
            "clientProvidedHostHeader": "guardduty-data.us-east-1.amazonaws.com"
        }
    }
```


The following example shows a CloudTrail log entry that demonstrates the `CreateIPThreatIntelSet` action (control plane event).


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

```
    "userIdentity": {
        "type": "AssumedRole",
        "principalId": "AIDACKCEVSQ6C2EXAMPLE",
        "arn": "arn:aws:iam::444455556666:user/Alice",
        "accountId": "444455556666",
        "accessKeyId": "AKIAI44QH8DHBEXAMPLE",
        "sessionContext": {
            "attributes": {
                "mfaAuthenticated": "false",
                "creationDate": "2018-06-14T22:54:20Z"
            },
            "sessionIssuer": {
                "type": "Role",
                "principalId": "AIDACKCEVSQ6C2EXAMPLE",
                "arn": "arn:aws:iam::444455556666:user/Alice",
                "accountId": "444455556666",
                "userName": "Alice"
            }
        }
    },
    "eventTime": "2018-06-14T22:57:56Z",
    "eventSource": "guardduty.amazonaws.com",
    "eventName": "CreateThreatIntelSet",
    "awsRegion": "us-west-2",
    "sourceIPAddress": "54.240.230.177",
    "userAgent": "console.amazonaws.com",
    "requestParameters": {
        "detectorId": "12abc34d567e8fa901bc2d34e56789f0",
        "name": "Example",
        "format": "TXT",
        "activate": false,
        "location": "https://s3.amazonaws.com/bucket.name/file.txt"
    },
    "responseElements": {
        "threatIntelSetId": "1ab200428351c99d859bf61992460d24"
    },
    "requestID": "5f6bf981-7026-11e8-a9fc-5b37d2684c5c",
    "eventID": "81337b11-e5c8-4f91-b141-deb405625bc9",
    "readOnly": false,
    "eventType": "AwsApiCall",
    "recipientAccountId": "444455556666"
}
```

From this event information, you can determine that the request was made to create a threat list `Example` in GuardDuty. You can also see that the request was made by a user named Alice on June 14, 2018.

Identity and Access Management for Amazon GuardDuty

AWS Identity and Access Management (IAM) is an AWS service that helps an administrator securely control access to AWS resources. IAM administrators control who can be *authenticated* (signed in) and *authorized* (have permissions) to use GuardDuty resources. IAM is an AWS service that you can use with no additional charge.

Topics

- Audience
- Authenticating with identities
- Managing access using policies
- How Amazon GuardDuty works with IAM
- Identity-based policy examples for Amazon GuardDuty
- Using service-linked roles for Amazon GuardDuty
- AWS managed policies for Amazon GuardDuty
- Troubleshooting Amazon GuardDuty identity and access

Audience

How you use AWS Identity and Access Management (IAM) differs, depending on the work that you do in GuardDuty.

Service user – If you use the GuardDuty service to do your job, then your administrator provides you with the credentials and permissions that you need. As you use more GuardDuty features to do your work, you might need additional permissions. Understanding how access is managed can help you request the right permissions from your administrator. If you cannot access a feature in GuardDuty, see Troubleshooting Amazon GuardDuty identity and access.

Service administrator – If you're in charge of GuardDuty resources at your company, you probably have full access to GuardDuty. It's your job to determine which GuardDuty features and resources your service users should access. You must then submit requests to your IAM administrator to change the permissions of your service users. Review the information on this page to understand the basic concepts of IAM. To learn more about how your company can use IAM with GuardDuty, see How Amazon GuardDuty works with IAM.

IAM administrator – If you're an IAM administrator, you might want to learn details about how you can write policies to manage access to GuardDuty. To view example GuardDuty identity-based policies that you can use in IAM, see Identity-based policy examples for Amazon GuardDuty.

Authenticating with identities

Authentication is how you sign in to AWS using your identity credentials. You must be *authenticated* (signed in to AWS) as the AWS account root user, as an IAM user, or by assuming an IAM role.

You can sign in to AWS as a federated identity by using credentials provided through an identity source. AWS IAM Identity Center (IAM Identity Center) users, your company's single sign-on authentication, and your Google or Facebook credentials are examples of federated identities. When you sign in as a federated identity, your administrator previously set up identity federation using IAM roles. When you access AWS by using federation, you are indirectly assuming a role.

Depending on the type of user you are, you can sign in to the AWS Management Console or the AWS access portal. For more information about signing in to AWS, see How to sign in to your AWS account in the *AWS Sign-In User Guide*.

If you access AWS programmatically, AWS provides a software development kit (SDK) and a command line interface (CLI) to cryptographically sign your requests by using your credentials. If you don't use AWS tools, you must sign requests yourself. For more information about using the recommended method to sign requests yourself, see AWS Signature Version 4 for API requests in the *IAM User Guide*.

Regardless of the authentication method that you use, you might be required to provide additional security information. For example, AWS recommends that you use multi-factor authentication (MFA) to increase the security of your account. To learn more, see Multi-factor authentication in the *AWS IAM Identity Center User Guide* and AWS Multi-factor authentication in IAM in the *IAM User Guide*.

AWS account root user

When you create an AWS account, you begin with one sign-in identity that has complete access to all AWS services and resources in the account. This identity is called the AWS account *root user* and is accessed by signing in with the email address and password that you used to create the account. We strongly recommend that you don't use the root user for your everyday tasks. Safeguard your root user credentials and use them to perform the tasks that only the root user can perform. For the complete list of tasks that require you to sign in as the root user, see Tasks that require root user credentials in the *IAM User Guide*.

Federated identity

As a best practice, require human users, including users that require administrator access, to use federation with an identity provider to access AWS services by using temporary credentials.

A *federated identity* is a user from your enterprise user directory, a web identity provider, the AWS Directory Service, the Identity Center directory, or any user that accesses AWS services by using credentials provided through an identity source. When federated identities access AWS accounts, they assume roles, and the roles provide temporary credentials.

For centralized access management, we recommend that you use AWS IAM Identity Center. You can create users and groups in IAM Identity Center, or you can connect and synchronize to a set of users and groups in your own identity source for use across all your AWS accounts and applications. For information about IAM Identity Center, see What is IAM Identity Center? in the *AWS IAM Identity Center User Guide*.

IAM users and groups

An *IAM user* is an identity within your AWS account that has specific permissions for a single person or application. Where possible, we recommend relying on temporary credentials instead of creating IAM users who have long-term credentials such as passwords and access keys. However, if you have specific use cases that require long-term credentials with IAM users, we recommend that you rotate access keys. For more information, see Rotate access keys regularly for use cases that require long-term credentials in the *IAM User Guide*.

An *IAM group* is an identity that specifies a collection of IAM users. You can't sign in as a group. You can use groups to specify permissions for multiple users at a time. Groups make permissions easier to manage for large sets of users. For example, you could have a group named *IAMAdmins* and give that group permissions to administer IAM resources.

Users are different from roles. A user is uniquely associated with one person or application, but a role is intended to be assumable by anyone who needs it. Users have permanent long-term credentials, but roles provide temporary credentials. To learn more, see Use cases for IAM users in the *IAM User Guide*.

IAM roles

An *IAM role* is an identity within your AWS account that has specific permissions. It is similar to an IAM user, but is not associated with a specific person. To temporarily assume an IAM role in the AWS Management Console, you can switch from a user to an IAM role (console). You can assume a role by calling an AWS CLI or AWS API operation or by using a custom URL. For more information about methods for using roles, see Methods to assume a role in the *IAM User Guide*.

IAM roles with temporary credentials are useful in the following situations:

- **Federated user access** – To assign permissions to a federated identity, you create a role and define permissions for the role. When a federated identity authenticates, the identity is associated with the role and is granted the permissions that are defined by the role. For information about roles for federation, see Create a role for a third-party identity provider (federation) in the *IAM User Guide*. If you use IAM Identity Center, you configure a permission set. To control what your identities can access after they authenticate, IAM Identity Center correlates the permission set to a role in IAM. For information about permissions sets, see Permission sets in the *AWS IAM Identity Center User Guide*.
- **Temporary IAM user permissions** – An IAM user or role can assume an IAM role to temporarily take on different permissions for a specific task.
- **Cross-account access** – You can use an IAM role to allow someone (a trusted principal) in a different account to access resources in your account. Roles are the primary way to grant cross-account access. However, with some AWS services, you can attach a policy directly to a resource (instead of using a role as a proxy). To learn the difference between roles and resource-based policies for cross-account access, see Cross account resource access in IAM in the *IAM User Guide*.
- **Cross-service access** – Some AWS services use features in other AWS services. For example, when you make a call in a service, it's common for that service to run applications in Amazon EC2 or store objects in Amazon S3. A service might do this using the calling principal's permissions, using a service role, or using a service-linked role.
  - **Forward access sessions (FAS)** – When you use an IAM user or role to perform actions in AWS, you are considered a principal. When you use some services, you might perform an action that then initiates another action in a different service. FAS uses the permissions of the principal calling an AWS service, combined with the requesting AWS service to make requests to downstream services. FAS requests are only made when a service receives a request that requires interactions with other AWS services or resources to complete. In this case, you must have permissions to perform both actions. For policy details when making FAS requests, see Forward access sessions.

  - **Service role –** A service role is an IAM role that a service assumes to perform actions on your behalf. An IAM administrator can create, modify, and delete a service role from within IAM. For more information, see Create a role to delegate permissions to an AWS service in the *IAM User Guide*.
  - **Service-linked role –** A service-linked role is a type of service role that is linked to an AWS service. The service can assume the role to perform an action on your behalf. Service-linked roles appear in your AWS account and are owned by the service. An IAM administrator can view, but not edit the permissions for service-linked roles.
- **Applications running on Amazon EC2 –** You can use an IAM role to manage temporary credentials for applications that are running on an EC2 instance and making AWS CLI or AWS API requests. This is preferable to storing access keys within the EC2 instance. To assign an AWS role to an EC2 instance and make it available to all of its applications, you create an instance profile that is attached to the instance. An instance profile contains the role and enables programs that are running on the EC2 instance to get temporary credentials. For more information, see Use an IAM role to grant permissions to applications running on Amazon EC2 instances in the *IAM User Guide*.

Managing access using policies

You control access in AWS by creating policies and attaching them to AWS identities or resources. A policy is an object in AWS that, when associated with an identity or resource, defines their permissions. AWS evaluates these policies when a principal (user, root user, or role session) makes a request. Permissions in the policies determine whether the request is allowed or denied. Most policies are stored in AWS as JSON documents. For more information about the structure and contents of JSON policy documents, see Overview of JSON policies in the *IAM User Guide*.

Administrators can use AWS JSON policies to specify who has access to what. That is, which **principal** can perform **actions** on what **resources**, and under what **conditions**.

By default, users and roles have no permissions. To grant users permission to perform actions on the resources that they need, an IAM administrator can create IAM policies. The administrator can then add the IAM policies to roles, and users can assume the roles.

IAM policies define permissions for an action regardless of the method that you use to perform the operation. For example, suppose that you have a policy that allows the `iam:GetRole` action. A user with that policy can get role information from the AWS Management Console, the AWS CLI, or the AWS API.

Identity-based policies

Identity-based policies are JSON permissions policy documents that you can attach to an identity, such as an IAM user, group of users, or role. These policies control what actions users and roles can perform, on which resources, and under what conditions. To learn how to create an identity-based policy, see Define custom IAM permissions with customer managed policies in the *IAM User Guide*.

Identity-based policies can be further categorized as *inline policies* or *managed policies*. Inline policies are embedded directly into a single user, group, or role. Managed policies are standalone policies that you can attach to multiple users, groups, and roles in your AWS account. Managed policies include AWS managed policies and customer managed policies. To learn how to choose between a managed policy or an inline policy, see Choose between managed policies and inline policies in the *IAM User Guide*.

Resource-based policies

Resource-based policies are JSON policy documents that you attach to a resource. Examples of resource-based policies are IAM *role trust policies* and Amazon S3 *bucket policies*. In services that support resource-based policies, service administrators can use them to control access to a specific resource. For the resource where the policy is attached, the policy defines what actions a specified principal can perform on that resource and under what conditions. You must specify a principal in a resource-based policy. Principals can include accounts, users, roles, federated users, or AWS services.

Resource-based policies are inline policies that are located in that service. You can't use AWS managed policies from IAM in a resource-based policy.

Access control lists (ACLs)

Access control lists (ACLs) control which principals (account members, users, or roles) have permissions to access a resource. ACLs are similar to resource-based policies, although they do not use the JSON policy document format.

Amazon S3, AWS WAF, and Amazon VPC are examples of services that support ACLs. To learn more about ACLs, see Access control list (ACL) overview in the *Amazon Simple Storage Service Developer Guide*.

Other policy types

AWS supports additional, less-common policy types. These policy types can set the maximum permissions granted to you by the more common policy types.

- **Permissions boundaries** – A permissions boundary is an advanced feature in which you set the maximum permissions that an identity-based policy can grant to an IAM entity (IAM user or role). You can set a permissions boundary for an entity. The resulting permissions are the intersection of an entity's identity-based policies and its permissions boundaries. Resource-based policies that specify the user or role in the `Principal` field are not limited by the permissions boundary. An explicit deny in any of these policies overrides the allow. For more information about permissions boundaries, see Permissions boundaries for IAM entities in the *IAM User Guide*.
- **Service control policies (SCPs)** – SCPs are JSON policies that specify the maximum permissions for an organization or organizational unit (OU) in AWS Organizations. AWS Organizations is a service for grouping and centrally managing multiple AWS accounts that your business owns. If you enable all features in an organization, then you can apply service control policies (SCPs) to any or all of your accounts. The SCP limits permissions for entities in member accounts, including each AWS account root user. For more information about Organizations and SCPs, see Service control policies in the *AWS Organizations User Guide*.
- **Resource control policies (RCPs)** – RCPs are JSON policies that you can use to set the maximum available permissions for resources in your accounts without updating the IAM policies attached to each resource that you own. The RCP limits permissions for resources in member accounts and can impact the effective permissions for identities, including the AWS account root user, regardless of whether they belong to your organization. For more information about Organizations and RCPs, including a list of AWS services that support RCPs, see Resource control policies (RCPs) in the *AWS Organizations User Guide*.
- **Session policies** – Session policies are advanced policies that you pass as a parameter when you programmatically create a temporary session for a role or federated user. The resulting session's permissions are the intersection of the user or role's identity-based policies and the session policies. Permissions can also come from a resource-based policy. An explicit deny in any of these policies overrides the allow. For more information, see Session policies in the *IAM User Guide*.

Multiple policy types

When multiple types of policies apply to a request, the resulting permissions are more complicated to understand. To learn how AWS determines whether to allow a request when multiple policy types are involved, see Policy evaluation logic in the *IAM User Guide*.

How Amazon GuardDuty works with IAM

Before you use IAM to manage access to GuardDuty, learn what IAM features are available to use with GuardDuty.

IAM features you can use with Amazon GuardDuty

IAM feature	GuardDuty support
Identity-based policies	Yes
Resource-based policies	No
Policy actions	Yes
Policy resources	Yes
Policy condition keys	Yes
ACLs	No
ABAC (tags in policies)	Partial
Temporary credentials	Yes
Principal permissions	Yes
Service roles	Yes
Service-linked roles	Yes

To get a high-level view of how GuardDuty and other AWS services work with most IAM features, see AWS services that work with IAM in the *IAM User Guide*.

Identity-based policies for GuardDuty

Supports identity-based policies: Yes

Identity-based policies are JSON permissions policy documents that you can attach to an identity, such as an IAM user, group of users, or role. These policies control what actions users and roles can perform, on which resources, and under what conditions. To learn how to create an identity-based policy, see Define custom IAM permissions with customer managed policies in the *IAM User Guide*.

With IAM identity-based policies, you can specify allowed or denied actions and resources as well as the conditions under which actions are allowed or denied. You can't specify the principal in an identity-based policy because it applies to the user or role to which it is attached. To learn about all of the elements that you can use in a JSON policy, see IAM JSON policy elements reference in the *IAM User Guide*.

Identity-based policy examples for GuardDuty

To view examples of GuardDuty identity-based policies, see Identity-based policy examples for Amazon GuardDuty.

Resource-based policies within GuardDuty

Supports resource-based policies: No

Resource-based policies are JSON policy documents that you attach to a resource. Examples of resource-based policies are IAM *role trust policies* and Amazon S3 *bucket policies*. In services that support resource-based policies, service administrators can use them to control access to a specific resource. For the resource where the policy is attached, the policy defines what actions a specified principal can perform on that resource and under what conditions. You must specify a principal in a resource-based policy. Principals can include accounts, users, roles, federated users, or AWS services.

To enable cross-account access, you can specify an entire account or IAM entities in another account as the principal in a resource-based policy. Adding a cross-account principal to a resource-based policy is only half of establishing the trust relationship. When the principal and the resource are in different AWS accounts, an IAM administrator in the trusted account must also grant the principal entity (user or role) permission to access the resource. They grant permission by attaching an identity-based policy to the entity. However, if a resource-based policy grants access to a principal in the same account, no additional identity-based policy is required. For more information, see Cross account resource access in IAM in the *IAM User Guide*.

Policy actions for GuardDuty

Supports policy actions: Yes

Administrators can use AWS JSON policies to specify who has access to what. That is, which **principal** can perform **actions** on what **resources**, and under what **conditions**.

The `Action` element of a JSON policy describes the actions that you can use to allow or deny access in a policy. Policy actions usually have the same name as the associated AWS API operation. There are some exceptions, such as *permission-only actions* that don't have a matching API operation. There are also some operations that require multiple actions in a policy. These additional actions are called *dependent actions*.

Include actions in a policy to grant permissions to perform the associated operation.

To see a list of GuardDuty actions, see Actions defined by Amazon GuardDuty in the *Service Authorization Reference*.

Policy actions in GuardDuty use the following prefix before the action:

```
guardduty
```

To specify multiple actions in a single statement, separate them with commas.

```
"Action": [
      "guardduty:action1",
      "guardduty:action2"
         ]
```

To view examples of GuardDuty identity-based policies, see Identity-based policy examples for Amazon GuardDuty.

Policy resources for GuardDuty

Supports policy resources: Yes

Administrators can use AWS JSON policies to specify who has access to what. That is, which **principal** can perform **actions** on what **resources**, and under what **conditions**.

The `Resource` JSON policy element specifies the object or objects to which the action applies. Statements must include either a `Resource` or a `NotResource` element. As a best practice, specify a resource using its Amazon Resource Name (ARN). You can do this for actions that support a specific resource type, known as *resource-level permissions*.

For actions that don't support resource-level permissions, such as listing operations, use a wildcard (*) to indicate that the statement applies to all resources.

```
"Resource": "*"
```

To see a list of GuardDuty resource types and their ARNs, see Resources defined by Amazon GuardDuty in the *Service Authorization Reference*. To learn with which actions you can specify the ARN of each resource, see Actions defined by Amazon GuardDuty.

To view examples of GuardDuty identity-based policies, see Identity-based policy examples for Amazon GuardDuty.

Policy condition keys for GuardDuty

Supports service-specific policy condition keys: Yes

Administrators can use AWS JSON policies to specify who has access to what. That is, which **principal** can perform **actions** on what **resources**, and under what **conditions**.

The `Condition` element (or `Condition` *block*) lets you specify conditions in which a statement is in effect. The `Condition` element is optional. You can create conditional expressions that use condition operators, such as equals or less than, to match the condition in the policy with values in the request.

If you specify multiple `Condition` elements in a statement, or multiple keys in a single `Condition` element, AWS evaluates them using a logical `AND` operation. If you specify multiple values for a single condition key, AWS evaluates the condition using a logical `OR` operation. All of the conditions must be met before the statement's permissions are granted.

You can also use placeholder variables when you specify conditions. For example, you can grant an IAM user permission to access a resource only if it is tagged with their IAM user name. For more information, see IAM policy elements: variables and tags in the *IAM User Guide*.

AWS supports global condition keys and service-specific condition keys. To see all AWS global condition keys, see AWS global condition context keys in the *IAM User Guide*.

To see a list of GuardDuty condition keys, see Condition keys for Amazon GuardDuty in the *Service Authorization Reference*. To learn with which actions and resources you can use a condition key, see Actions defined by Amazon GuardDuty.

To view examples of GuardDuty identity-based policies, see Identity-based policy examples for Amazon GuardDuty.

Access control lists (ACLs) in GuardDuty

Supports ACLs: No

Access control lists (ACLs) control which principals (account members, users, or roles) have permissions to access a resource. ACLs are similar to resource-based policies, although they do not use the JSON policy document format.

Attribute-based access control (ABAC) with GuardDuty

Supports ABAC (tags in policies): Partial

Attribute-based access control (ABAC) is an authorization strategy that defines permissions based on attributes. In AWS, these attributes are called *tags*. You can attach tags to IAM entities (users or roles) and to many AWS resources. Tagging entities and resources is the first step of ABAC. Then you design ABAC policies to allow operations when the principal's tag matches the tag on the resource that they are trying to access.

ABAC is helpful in environments that are growing rapidly and helps with situations where policy management becomes cumbersome.

To control access based on tags, you provide tag information in the condition element of a policy using the `aws:ResourceTag/key-name`, `aws:RequestTag/key-name`, or `aws:TagKeys` condition keys.

If a service supports all three condition keys for every resource type, then the value is **Yes** for the service. If a service supports all three condition keys for only some resource types, then the value is **Partial**.

For more information about ABAC, see Define permissions with ABAC authorization in the *IAM User Guide*. To view a tutorial with steps for setting up ABAC, see Use attribute-based access control (ABAC) in the *IAM User Guide*.

Using Temporary credentials with GuardDuty

Supports temporary credentials: Yes

Some AWS services don't work when you sign in using temporary credentials. For additional information, including which AWS services work with temporary credentials, see AWS services that work with IAM in the *IAM User Guide*.

You are using temporary credentials if you sign in to the AWS Management Console using any method except a user name and password. For example, when you access AWS using your company's single sign-on (SSO) link, that process automatically creates temporary credentials. You also automatically create temporary credentials when you sign in to the console as a user and then switch roles. For more information about switching roles, see Switch from a user to an IAM role (console) in the *IAM User Guide*.

You can manually create temporary credentials using the AWS CLI or AWS API. You can then use those temporary credentials to access AWS. AWS recommends that you dynamically generate temporary credentials instead of using long-term access keys. For more information, see Temporary security credentials in IAM.

Cross-service principal permissions for GuardDuty

Supports forward access sessions (FAS): Yes

When you use an IAM user or role to perform actions in AWS, you are considered a principal. When you use some services, you might perform an action that then initiates another action in a different service. FAS uses the permissions of the principal calling an AWS service, combined with the requesting AWS service to make requests to downstream services. FAS requests are only made when a service receives a request that requires interactions with other AWS services or resources to complete. In this case, you must have permissions to perform both actions. For policy details when making FAS requests, see Forward access sessions.

Service roles for GuardDuty

Supports service roles: Yes

A service role is an IAM role that a service assumes to perform actions on your behalf. An IAM administrator can create, modify, and delete a service role from within IAM. For more information, see Create a role to delegate permissions to an AWS service in the *IAM User Guide*.

> **Warning**
>
> Changing the permissions for a service role might break GuardDuty functionality. Edit service roles only when GuardDuty provides guidance to do so.

Service-linked roles for GuardDuty

Supports service-linked roles: Yes

A service-linked role is a type of service role that is linked to an AWS service. The service can assume the role to perform an action on your behalf. Service-linked roles appear in your AWS account and are owned by the service. An IAM administrator can view, but not edit the permissions for service-linked roles.

For details about creating or managing GuardDuty service-linked roles, see Using service-linked roles for Amazon GuardDuty.

For details about creating or managing service-linked roles, see AWS services that work with IAM. Find a service in the table that includes a `Yes` in the **Service-linked role** column. Choose the **Yes** link to view the service-linked role documentation for that service.

Identity-based policy examples for Amazon GuardDuty

By default, users and roles don't have permission to create or modify GuardDuty resources. They also can't perform tasks by using the AWS Management Console, AWS Command Line Interface (AWS CLI), or AWS API. To grant users permission to perform actions on the resources that they need, an IAM administrator can create IAM policies. The administrator can then add the IAM policies to roles, and users can assume the roles.

To learn how to create an IAM identity-based policy by using these example JSON policy documents, see Create IAM policies (console) in the *IAM User Guide*.

For details about actions and resource types defined by GuardDuty, including the format of the ARNs for each of the resource types, see Actions, resources, and condition keys for Amazon GuardDuty in the *Service Authorization Reference*.

Topics

- Policy best practices

- Using the GuardDuty console
- Permissions required to enable GuardDuty
- Allow users to view their own permissions
- Custom IAM policy to grant read-only access to GuardDuty
- Deny Access to GuardDuty findings
- Using a custom IAM policy to limit access to GuardDuty resources

Policy best practices

Identity-based policies determine whether someone can create, access, or delete GuardDuty resources in your account. These actions can incur costs for your AWS account. When you create or edit identity-based policies, follow these guidelines and recommendations:

- **Get started with AWS managed policies and move toward least-privilege permissions** – To get started granting permissions to your users and workloads, use the *AWS managed policies* that grant permissions for many common use cases. They are available in your AWS account. We recommend that you reduce permissions further by defining AWS customer managed policies that are specific to your use cases. For more information, see AWS managed policies or AWS managed policies for job functions in the *IAM User Guide*.
- **Apply least-privilege permissions** – When you set permissions with IAM policies, grant only the permissions required to perform a task. You do this by defining the actions that can be taken on specific resources under specific conditions, also known as *least-privilege permissions*. For more information about using IAM to apply permissions, see Policies and permissions in IAM in the *IAM User Guide*.
- **Use conditions in IAM policies to further restrict access** – You can add a condition to your policies to limit access to actions and resources. For example, you can write a policy condition to specify that all requests must be sent using SSL. You can also use conditions to grant access to service actions if they are used through a specific AWS service, such as AWS CloudFormation. For more information, see IAM JSON policy elements: Condition in the *IAM User Guide*.
- **Use IAM Access Analyzer to validate your IAM policies to ensure secure and functional permissions** – IAM Access Analyzer validates new and existing policies so that the policies adhere to the IAM policy language (JSON) and IAM best practices. IAM Access Analyzer provides more than 100 policy checks and actionable recommendations to help you author secure and functional policies. For more information, see Validate policies with IAM Access Analyzer in the *IAM User Guide*.

- **Require multi-factor authentication (MFA)** – If you have a scenario that requires IAM users or a root user in your AWS account, turn on MFA for additional security. To require MFA when API operations are called, add MFA conditions to your policies. For more information, see Secure API access with MFA in the *IAM User Guide*.

For more information about best practices in IAM, see Security best practices in IAM in the *IAM User Guide*.

Using the GuardDuty console

To access the Amazon GuardDuty console, you must have a minimum set of permissions. These permissions must allow you to list and view details about the GuardDuty resources in your AWS account. If you create an identity-based policy that is more restrictive than the minimum required permissions, the console won't function as intended for entities (users or roles) with that policy.

You don't need to allow minimum console permissions for users that are making calls only to the AWS CLI or the AWS API. Instead, allow access to only the actions that match the API operation that they're trying to perform.

To ensure that users and roles can still use the GuardDuty console, also attach the GuardDuty `ConsoleAccess` or `ReadOnly` AWS managed policy to the entities. For more information, see Adding permissions to a user in the *IAM User Guide*.

Permissions required to enable GuardDuty

To grant permissions that various IAM identities (users, groups, and roles) must have, attach the required AWS managed policy: AmazonGuardDutyFullAccess_v2 (recommended) policy to enable GuardDuty.

Allow users to view their own permissions

This example shows how you might create a policy that allows IAM users to view the inline and managed policies that are attached to their user identity. This policy includes permissions to complete this action on the console or programmatically using the AWS CLI or AWS API.

```
{
    "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": "*"
        }
    ]
}
```

Custom IAM policy to grant read-only access to GuardDuty

To grant read-only access to GuardDuty you can use the `AmazonGuardDutyReadOnlyAccess` managed policy.

To create a custom policy that grants an IAM role, user, or group read-only access to GuardDuty, you can use the following statement:

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
```

```
            "Effect": "Allow",
            "Action": [
                "guardduty:ListMembers",
                "guardduty:GetMembers",
                "guardduty:ListInvitations",
                "guardduty:ListDetectors",
                "guardduty:GetDetector",
                "guardduty:ListFindings",
                "guardduty:GetFindings",
                "guardduty:ListIPSets",
                "guardduty:GetIPSet",
                "guardduty:ListThreatIntelSets",
                "guardduty:GetThreatIntelSet",
                "guardduty:GetMasterAccount",
                "guardduty:GetInvitationsCount",
                "guardduty:GetFindingsStatistics",
                "guardduty:DescribeMalwareScans",
                "guardduty:UpdateMalwareScanSettings",
                "guardduty:GetMalwareScanSettings"
            ],
            "Resource": "*"
        }
    ]
}
```

Deny Access to GuardDuty findings

You can use the following policy to deny an IAM role, user, or group access to GuardDuty findings. Users can't view findings or the details about findings, but they can access all other GuardDuty operations:

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "guardduty:CreateDetector",
                "guardduty:DeleteDetector",
```

```
                "guardduty:UpdateDetector",
                "guardduty:GetDetector",
                "guardduty:ListDetectors",
                "guardduty:CreateIPSet",
                "guardduty:DeleteIPSet",
                "guardduty:UpdateIPSet",
                "guardduty:GetIPSet",
                "guardduty:ListIPSets",
                "guardduty:CreateThreatIntelSet",
                "guardduty:DeleteThreatIntelSet",
                "guardduty:UpdateThreatIntelSet",
                "guardduty:GetThreatIntelSet",
                "guardduty:ListThreatIntelSets",
                "guardduty:ArchiveFindings",
                "guardduty:UnarchiveFindings",
                "guardduty:CreateSampleFindings",
                "guardduty:CreateMembers",
                "guardduty:InviteMembers",
                "guardduty:GetMembers",
                "guardduty:DeleteMembers",
                "guardduty:DisassociateMembers",
                "guardduty:StartMonitoringMembers",
                "guardduty:StopMonitoringMembers",
                "guardduty:ListMembers",
                "guardduty:GetMasterAccount",
                "guardduty:DisassociateFromMasterAccount",
                "guardduty:AcceptAdministratorInvitation",
                "guardduty:ListInvitations",
                "guardduty:GetInvitationsCount",
                "guardduty:DeclineInvitations",
                "guardduty:DeleteInvitations"
            ],
            "Resource": "*"
        },
         {
            "Effect": "Allow",
            "Action": [
                "iam:CreateServiceLinkedRole"
            ],
            "Resource": "arn:aws:iam::123456789012:role/aws-service-role/
guardduty.amazonaws.com/AWSServiceRoleForAmazonGuardDuty",
            "Condition": {
                "StringLike": {
                    "iam:AWSServiceName": "guardduty.amazonaws.com"
```

```
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": [
                "iam:PutRolePolicy",
                "iam:DeleteRolePolicy"
            ],
            "Resource": "arn:aws:iam::123456789012:role/aws-service-role/
guardduty.amazonaws.com/AWSServiceRoleForAmazonGuardDuty"
        }
    ]
}
```

Using a custom IAM policy to limit access to GuardDuty resources

To define a user's access to GuardDuty based on the detector ID, you can use all GuardDuty API actions in your custom IAM policies, **except** the following operations:

- `guardduty:CreateDetector`
- `guardduty:DeclineInvitations`
- `guardduty:DeleteInvitations`
- `guardduty:GetInvitationsCount`
- `guardduty:ListDetectors`
- `guardduty:ListInvitations`

Use the following operations in an IAM policy to define a user's access to GuardDuty based on the IPSet ID and ThreatIntelSet ID:

- `guardduty:DeleteIPSet`
- `guardduty:DeleteThreatIntelSet`
- `guardduty:GetIPSet`
- `guardduty:GetThreatIntelSet`
- `guardduty:UpdateIPSet`
- `guardduty:UpdateThreatIntelSet`

The following examples show how to create policies using some of the preceding operations:

- This policy allows a user to run the `guardduty:UpdateDetector` operation, using the detector ID of 1234567 in the us-east-1 Region:

  JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "guardduty:UpdateDetector"
            ],
            "Resource": "arn:aws:guardduty:us-
east-1:123456789012:detector/1234567"
        }
    ]
}
```

- This policy allows a user to run the `guardduty:UpdateIPSet` operation, using the detector ID of 1234567 and the IPSet ID of 000000 in the us-east-1 Region:

> **Note**
>
> Make sure that the user has the permissions required to access trusted IP lists and threat lists in GuardDuty. For more information, see Setting up prerequisites for entity lists and IP address lists.

  JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "guardduty:UpdateIPSet"
            ],
```

```
            "Resource": "arn:aws:guardduty:us-
east-1:123456789012:detector/1234567/ipset/000000"
        }
    ]
}
```

- This policy allows a user to run the `guardduty:UpdateIPSet` operation, using any detector ID and the IPSet ID of 000000 in the us-east-1 Region:

> **Note**
>
> Make sure that the user has the permissions required to access trusted IP lists and threat lists in GuardDuty. For more information, see [Setting up prerequisites for entity lists and IP address lists](#).

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "guardduty:UpdateIPSet"
            ],
            "Resource": "arn:aws:guardduty:us-east-1:123456789012:detector/*/
ipset/000000"
        }
    ]
}
```

- This policy allows a user to run the `guardduty:UpdateIPSet` operation, using their detector ID and any IPSet ID in the us-east-1 Region:

> **Note**
>
> Make sure that the user has the permissions required to access trusted IP lists and threat lists in GuardDuty. For more information, see [Setting up prerequisites for entity lists and IP address lists](#).

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "guardduty:UpdateIPSet"
            ],
            "Resource": "arn:aws:guardduty:us-
east-1:123456789012:detector/1234567/ipset/*"
        }
    ]
}
```

Using service-linked roles for Amazon GuardDuty

Amazon GuardDuty uses AWS Identity and Access Management (IAM) service-linked roles. A service-linked role (SLR) is a unique type of IAM role that is linked directly to GuardDuty. Service-linked roles are predefined by GuardDuty and include all the permissions that GuardDuty requires to call other AWS services on your behalf.

With service-linked role, you can set up GuardDuty without adding the necessary permissions manually. GuardDuty defines the permissions of its service-linked role, and unless the permissions are defined otherwise, only GuardDuty can assume the role. The defined permissions include the trust policy and the permissions policy, and that permissions policy can't be attached to any other IAM entity.

GuardDuty supports using service-linked roles in all of the Regions where GuardDuty is available. For more information, see Regions and endpoints.

You can delete the GuardDuty service-linked role only after first disabling GuardDuty in all Regions where it is enabled. This protects your GuardDuty resources because you can't inadvertently remove permission to access them.

For information about other services that support service-linked roles, see AWS services that work with IAM in the *IAM User Guide* and look for the services that have **Yes** in the **Service-Linked Role** column. Choose a **Yes** with a link to view the service-linked role documentation for that service.

Service-linked role permissions for GuardDuty

GuardDuty uses the service-linked role (SLR) named `AWSServiceRoleForAmazonGuardDuty`. The SLR allows GuardDuty to perform the following tasks. It also allows GuardDuty to include the retrieved metadata belonging to the EC2 instance in the findings that GuardDuty may generate about the potential threat. The `AWSServiceRoleForAmazonGuardDuty` service-linked role trusts the `guardduty.amazonaws.com` service to assume the role.

The permission policies help GuardDuty perform the following tasks:

- Use Amazon EC2 actions to manage and retrieve information about your EC2 instances, images, and networking components such as VPCs, subnets, and transit gateways.
- Use AWS Systems Manager actions to manage SSM associations on Amazon EC2 instances when you enable GuardDuty Runtime Monitoring with automated agent for Amazon EC2. When GuardDuty automated agent configuration is disabled, GuardDuty considers only those EC2 instances that have an inclusion tag (`GuardDutyManaged`:`true`).
- Use AWS Organizations actions to describe associated accounts and organization ID.
- Use Amazon S3 actions to retrieve information about S3 buckets and objects.
- Use AWS Lambda actions to retrieve information about your Lambda functions and tags.
- Use Amazon EKS actions to manage and retrieve information about the EKS clusters and manage Amazon EKS add-ons on EKS clusters. The EKS actions also retrieve the information about the tags associated to GuardDuty.
- Use IAM to create the Service-linked role permissions for Malware Protection for EC2 after Malware Protection for EC2 has been enabled.
- Use Amazon ECS actions to manage and retrieve information about the Amazon ECS clusters, and manage the Amazon ECS account setting with `guarddutyActivate`. The actions pertaining to Amazon ECS also retrieve the information about the tags associated with GuardDuty.

The role is configured with the following AWS managed policy, named `AmazonGuardDutyServiceRolePolicy`.

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "GuardDutyGetDescribeListPolicy",
            "Effect": "Allow",
            "Action": [
                "ec2:DescribeInstances",
                "ec2:DescribeImages",
                "ec2:DescribeVpcEndpoints",
                "ec2:DescribeSubnets",
                "ec2:DescribeVpcPeeringConnections",
                "ec2:DescribeTransitGatewayAttachments",
                "organizations:ListAccounts",
                "organizations:DescribeAccount",
                "organizations:DescribeOrganization",
                "s3:GetBucketPublicAccessBlock",
                "s3:GetEncryptionConfiguration",
                "s3:GetBucketTagging",
                "s3:GetAccountPublicAccessBlock",
                "s3:ListAllMyBuckets",
                "s3:GetBucketAcl",
                "s3:GetBucketPolicy",
                "s3:GetBucketPolicyStatus",
                "lambda:GetFunctionConfiguration",
                "lambda:ListTags",
                "eks:ListClusters",
                "eks:DescribeCluster",
                "ec2:DescribeVpcEndpointServices",
                "ec2:DescribeSecurityGroups",
                "ec2:DescribeVpcs",
                "ecs:ListClusters",
                "ecs:DescribeClusters"
            ],
            "Resource": "*"
        },
        {
            "Sid": "GuardDutyCreateSLRPolicy",
            "Effect": "Allow",
            "Action": "iam:CreateServiceLinkedRole",
            "Resource": "*",
```

```
            "Condition": {
                "StringEquals": {
                    "iam:AWSServiceName": "malware-
protection.guardduty.amazonaws.com"
                }
            }
        },
        {
            "Sid": "GuardDutyCreateVpcEndpointPolicy",
            "Effect": "Allow",
            "Action": "ec2:CreateVpcEndpoint",
            "Resource": "arn:aws:ec2:*:*:vpc-endpoint/*",
            "Condition": {
                "ForAnyValue:StringEquals": {
                    "aws:TagKeys": "GuardDutyManaged"
                },
                "StringLike": {
                    "ec2:VpceServiceName": [
                        "com.amazonaws.*.guardduty-data",
                        "com.amazonaws.*.guardduty-data-fips"
                    ]
                }
            }
        },
        {
            "Sid": "GuardDutyModifyDeleteVpcEndpointPolicy",
            "Effect": "Allow",
            "Action": [
                "ec2:ModifyVpcEndpoint",
                "ec2:DeleteVpcEndpoints"
            ],
            "Resource": "arn:aws:ec2:*:*:vpc-endpoint/*",
            "Condition": {
                "Null": {
                    "aws:ResourceTag/GuardDutyManaged": false
                }
            }
        },
        {
            "Sid": "GuardDutyCreateModifyVpcEndpointNetworkPolicy",
            "Effect": "Allow",
            "Action": [
                "ec2:CreateVpcEndpoint",
                "ec2:ModifyVpcEndpoint"
```

```
            ],
            "Resource": [
                "arn:aws:ec2:*:*:vpc/*",
                "arn:aws:ec2:*:*:security-group/*",
                "arn:aws:ec2:*:*:subnet/*"
            ]
        },
        {
             "Sid": "GuardDutyCreateTagsDuringVpcEndpointCreationPolicy",
             "Effect": "Allow",
            "Action": "ec2:CreateTags",
            "Resource": "arn:aws:ec2:*:*:vpc-endpoint/*",
            "Condition": {
                "StringEquals": {
                    "ec2:CreateAction": "CreateVpcEndpoint"
                },
                "ForAnyValue:StringEquals": {
                    "aws:TagKeys": "GuardDutyManaged"
                }
            }
        },
        {
            "Sid": "GuardDutySecurityGroupManagementPolicy",
            "Effect": "Allow",
            "Action": [
                "ec2:AuthorizeSecurityGroupIngress",
                "ec2:AuthorizeSecurityGroupEgress",
                "ec2:RevokeSecurityGroupIngress",
                "ec2:RevokeSecurityGroupEgress",
                "ec2:DeleteSecurityGroup"
            ],
            "Resource": "arn:aws:ec2:*:*:security-group/*",
            "Condition": {
                "Null": {
                    "aws:ResourceTag/GuardDutyManaged": false
                }
            }
        },
        {
            "Sid": "GuardDutyCreateSecurityGroupPolicy",
            "Effect": "Allow",
            "Action": "ec2:CreateSecurityGroup",
            "Resource": "arn:aws:ec2:*:*:security-group/*",
            "Condition": {
```

```
                "StringLike": {
                    "aws:RequestTag/GuardDutyManaged": "*"
                }
            }
        },
        {
            "Sid": "GuardDutyCreateSecurityGroupForVpcPolicy",
            "Effect": "Allow",
            "Action": "ec2:CreateSecurityGroup",
            "Resource": "arn:aws:ec2:*:*:vpc/*"
        },
        {
            "Sid": "GuardDutyCreateTagsDuringSecurityGroupCreationPolicy",
            "Effect": "Allow",
            "Action": "ec2:CreateTags",
            "Resource": "arn:aws:ec2:*:*:security-group/*",
            "Condition": {
                "StringEquals": {
                    "ec2:CreateAction": "CreateSecurityGroup"
                },
                "ForAnyValue:StringEquals": {
                    "aws:TagKeys": "GuardDutyManaged"
                }
            }
        },
        {
            "Sid": "GuardDutyCreateEksAddonPolicy",
            "Effect": "Allow",
            "Action": "eks:CreateAddon",
            "Resource": "arn:aws:eks:*:*:cluster/*",
            "Condition": {
                "ForAnyValue:StringEquals": {
                    "aws:TagKeys": "GuardDutyManaged"
                }
            }
        },
        {
            "Sid": "GuardDutyEksAddonManagementPolicy",
            "Effect": "Allow",
            "Action": [
                "eks:DeleteAddon",
                "eks:UpdateAddon",
                "eks:DescribeAddon"
            ],
```

```
            "Resource": "arn:aws:eks:*:*:addon/*/aws-guardduty-agent/*"
        },
        {
            "Sid": "GuardDutyEksClusterTagResourcePolicy",
            "Effect": "Allow",
            "Action": "eks:TagResource",
            "Resource": "arn:aws:eks:*:*:cluster/*",
            "Condition": {
                "ForAnyValue:StringEquals": {
                    "aws:TagKeys": "GuardDutyManaged"
                }
            }
        },
        {
            "Sid": "GuardDutyEcsPutAccountSettingsDefaultPolicy",
            "Effect": "Allow",
            "Action": "ecs:PutAccountSettingDefault",
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "ecs:account-setting": [
                        "guardDutyActivate"
                    ]
                 }
            }
        },
        {
            "Sid": "SsmCreateDescribeUpdateDeleteStartAssociationPermission",
            "Effect": "Allow",
            "Action": [
                "ssm:DescribeAssociation",
                "ssm:DeleteAssociation",
                "ssm:UpdateAssociation",
                "ssm:CreateAssociation",
                "ssm:StartAssociationsOnce"
            ],
            "Resource": "arn:aws:ssm:*:*:association/*",
            "Condition": {
                "StringEquals": {
                    "aws:ResourceTag/GuardDutyManaged": "true"
                }
            }
        },
        {
```

```
            "Sid": "SsmAddTagsToResourcePermission",
            "Effect": "Allow",
            "Action": [
                "ssm:AddTagsToResource"
            ],
            "Resource": "arn:aws:ssm:*:*:association/*",
            "Condition":{
                "ForAllValues:StringEquals": {
                    "aws:TagKeys": [
                        "GuardDutyManaged"
                    ]
                },
                "StringEquals": {
                    "aws:ResourceTag/GuardDutyManaged": "true"
                }
            }
        },
        {
            "Sid": "SsmCreateUpdateAssociationInstanceDocumentPermission",
            "Effect": "Allow",
            "Action": [
                "ssm:CreateAssociation",
                "ssm:UpdateAssociation"
            ],
            "Resource": "arn:aws:ssm:*:*:document/AmazonGuardDuty-
ConfigureRuntimeMonitoringSsmPlugin"
        },
        {
            "Sid": "SsmSendCommandPermission",
            "Effect": "Allow",
            "Action": "ssm:SendCommand",
            "Resource": [
                "arn:aws:ec2:*:*:instance/*",
                "arn:aws:ssm:*:*:document/AmazonGuardDuty-
ConfigureRuntimeMonitoringSsmPlugin"
            ]
        },
        {
            "Sid": "SsmGetCommandStatus",
            "Effect": "Allow",
            "Action": "ssm:GetCommandInvocation",
            "Resource": "*"
        }
    ]
```

```
}
```

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ec2:DescribeInstances",
                "ec2:DescribeImages",
                "ec2:DescribeVpcEndpoints",
                "ec2:DescribeSubnets",
                "ec2:DescribeVpcPeeringConnections",
                "ec2:DescribeTransitGatewayAttachments",
                "organizations:ListAccounts",
                "organizations:DescribeAccount",
                "s3:GetBucketPublicAccessBlock",
                "s3:GetEncryptionConfiguration",
                "s3:GetBucketTagging",
                "s3:GetAccountPublicAccessBlock",
                "s3:ListAllMyBuckets",
                "s3:GetBucketAcl",
                "s3:GetBucketPolicy",
                "s3:GetBucketPolicyStatus",
                "lambda:GetFunctionConfiguration",
                "lambda:ListTags"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": "iam:CreateServiceLinkedRole",
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "iam:AWSServiceName": "malware-
protection.guardduty.amazonaws.com"
                }
            }
        }
```

```
    ]
}
```

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ec2:DescribeInstances",
                "ec2:DescribeImages",
                "ec2:DescribeVpcEndpoints",
                "ec2:DescribeSubnets",
                "ec2:DescribeVpcPeeringConnections",
                "ec2:DescribeTransitGatewayAttachments",
                "organizations:ListAccounts",
                "organizations:DescribeAccount",
                "s3:GetBucketPublicAccessBlock",
                "s3:GetEncryptionConfiguration",
                "s3:GetBucketTagging",
                "s3:GetAccountPublicAccessBlock",
                "s3:ListAllMyBuckets",
                "s3:GetBucketAcl",
                "s3:GetBucketPolicy",
                "s3:GetBucketPolicyStatus",
                "lambda:GetFunctionConfiguration",
                "lambda:ListTags"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": "iam:CreateServiceLinkedRole",
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "iam:AWSServiceName": "malware-
protection.guardduty.amazonaws.com"
                }
            }
```

```
        }
    ]
}
```

JSON

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "GuardDutyGetDescribeListPolicy",
      "Effect": "Allow",
      "Action": [
        "ec2:DescribeInstances",
        "ec2:DescribeImages",
        "ec2:DescribeVpcEndpoints",
        "ec2:DescribeSubnets",
        "ec2:DescribeVpcPeeringConnections",
        "ec2:DescribeTransitGatewayAttachments",
        "organizations:ListAccounts",
        "organizations:DescribeAccount",
        "s3:GetBucketPublicAccessBlock",
        "s3:GetEncryptionConfiguration",
        "s3:GetBucketTagging",
        "s3:GetAccountPublicAccessBlock",
        "s3:ListAllMyBuckets",
        "s3:GetBucketAcl",
        "s3:GetBucketPolicy",
        "s3:GetBucketPolicyStatus",
        "lambda:GetFunctionConfiguration",
        "lambda:ListTags"
      ],
        "Resource": "*"
    },
    {
      "Sid": "GuardDutyCreateSLRPolicy",
      "Effect": "Allow",
      "Action": "iam:CreateServiceLinkedRole",
      "Resource": "*",
      "Condition": {
        "StringEquals": {
          "iam:AWSServiceName": "malware-protection.guardduty.amazonaws.com"
```

```
        }
      }
    }
  ]
}
```

The following is the trust policy that is attached to the `AWSServiceRoleForAmazonGuardDuty` service-linked role:

JSON

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "guardduty.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
```

For details about updates to the `AmazonGuardDutyServiceRolePolicy` policy, see GuardDuty updates to AWS managed policies. For automatic alerts about changes to this policy, subscribe to the RSS feed on the Document history page.

Creating a service-linked role for GuardDuty

The `AWSServiceRoleForAmazonGuardDuty` service-linked role is automatically created when you enable GuardDuty for the first time or enable GuardDuty in a supported Region where you previously didn't have it enabled. You can also create the service-linked role manually using the IAM console, the AWS CLI, or the IAM API.

Important

The service-linked role that is created for the GuardDuty delegated administrator account doesn't apply to the member GuardDuty accounts.

You must configure permissions to allow an IAM principal (such as a user, group, or role) to create, edit, or delete a service-linked role. For the `AWSServiceRoleForAmazonGuardDuty` service-linked role to be successfully created, the IAM principal that you use GuardDuty with must have the required permissions. To grant the required permissions, attach the following policy to this user, group, or role:

Note

Replace the sample *`account ID`* in the following example with your actual AWS account ID.

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "guardduty:*"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "iam:CreateServiceLinkedRole"
            ],
            "Resource": "arn:aws:iam::123456789012:role/aws-service-role/
guardduty.amazonaws.com/AWSServiceRoleForAmazonGuardDuty",
            "Condition": {
                "StringLike": {
                    "iam:AWSServiceName": "guardduty.amazonaws.com"
```

```
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": [
                "iam:PutRolePolicy",
                "iam:DeleteRolePolicy"
            ],
            "Resource": "arn:aws:iam::123456789012:role/aws-service-role/
guardduty.amazonaws.com/AWSServiceRoleForAmazonGuardDuty"
        }
    ]
}
```

For more information about creating the role manually, see Creating a service-linked role in the *IAM User Guide*.

Editing a service-linked role for GuardDuty

GuardDuty doesn't allow you to edit the `AWSServiceRoleForAmazonGuardDuty` service-linked role. After you create a service-linked role, you can't change the name of the role because various entities might reference the role. However, you can edit the description of the role using IAM. For more information, see Editing a service-linked role in the *IAM User Guide*.

Deleting a service-linked role for GuardDuty

If you no longer need to use a feature or service that requires a service-linked role, we recommend that you delete that role. That way you don't have an unused entity that isn't actively monitored or maintained.

> **Important**
>
> If you have enabled Malware Protection for EC2, deleting `AWSServiceRoleForAmazonGuardDuty` doesn't automatically delete `AWSServiceRoleForAmazonGuardDutyMalwareProtection`. If you want to delete `AWSServiceRoleForAmazonGuardDutyMalwareProtection`, see Deleting a service-linked role for Malware Protection for EC2.

You must first disable GuardDuty in all Regions where it is enabled in order to delete the `AWSServiceRoleForAmazonGuardDuty`. If the GuardDuty service isn't disabled when you try to delete the service-linked role, the deletion fails. For more information, see [Suspending or disabling GuardDuty](#).

When you disable GuardDuty, the `AWSServiceRoleForAmazonGuardDuty` doesn't get deleted automatically. If you enable GuardDuty again, it'll start using the existing `AWSServiceRoleForAmazonGuardDuty`.

To manually delete the service-linked role using IAM

Use the IAM console, the AWS CLI, or the IAM API to delete the `AWSServiceRoleForAmazonGuardDuty` service-linked role. For more information, see [Deleting a service-linked role](#) in the *IAM User Guide*.

Supported AWS Regions

Amazon GuardDuty supports using the `AWSServiceRoleForAmazonGuardDuty` service-linked role in all the AWS Regions where GuardDuty is available. For a list of Regions where GuardDuty is currently available, see [Amazon GuardDuty endpoints and quotas](#) in the *Amazon Web Services General Reference*.

Service-linked role permissions for Malware Protection for EC2

Malware Protection for EC2 uses the service-linked role (SLR) named `AWSServiceRoleForAmazonGuardDutyMalwareProtection`. This SLR allows Malware Protection for EC2 to perform agentless scans to detect malware in your GuardDuty account. It allows GuardDuty to create an EBS volume snapshot in your account, and share that snapshot with the GuardDuty service account. After GuardDuty evaluates the snapshot, it includes the retrieved EC2 instance and container workload metadata in the Malware Protection for EC2 findings. The `AWSServiceRoleForAmazonGuardDutyMalwareProtection` service-linked role trusts the `malware-protection.guardduty.amazonaws.com` service to assume the role.

The permission policies for this role helps Malware Protection for EC2 to perform the following tasks:

- Use Amazon Elastic Compute Cloud (Amazon EC2) actions to retrieve information about your Amazon EC2 instances, volumes, and snapshots. Malware Protection for EC2 also provides permission to access the Amazon EKS and Amazon ECS cluster metadata.

- Create snapshots for EBS volumes that have `GuardDutyExcluded` tag not set to `true`. By default, the snapshots get created with a `GuardDutyScanId` tag. Don't remove this tag, otherwise Malware Protection for EC2 will not have access to the snapshots.

> **Important**
>
> When you set the `GuardDutyExcluded` to `true`, the GuardDuty service won't be able to access these snapshots in the future. This is because the other statements in this service-linked role prevent GuardDuty from performing any action on the snapshots that have the `GuardDutyExcluded` set to `true`.

- Allow sharing and deleting snapshots only if the `GuardDutyScanId` tag exists and `GuardDutyExcluded` tag is not set to `true`.

> **Note**
>
> Doesn't allow Malware Protection for EC2 to make the snapshots public.

- Access customer managed keys, except those that have a `GuardDutyExcluded` tag set to `true`, to call `CreateGrant` to create and access an encrypted EBS volume from the encrypted snapshot that gets shared with the GuardDuty service account. For a list of GuardDuty service accounts for each Region, see [GuardDuty service accounts by AWS Region].
- Access customers' CloudWatch logs to create the Malware Protection for EC2 log group as well as put the malware scan events logs under the `/aws/guardduty/malware-scan-events` log group.
- Allow the customer to decide if they want to keep the snapshots on which malware was detected, in their account. If the scan detects malware, the service-linked role allows GuardDuty to add two tags to snapshots - `GuardDutyFindingDetected` and `GuardDutyExcluded`.

> **Note**
>
> The `GuardDutyFindingDetected` tag specifies that the snapshots contains malware.

- Determine if a volume is encrypted with an EBS managed key. GuardDuty performs the `DescribeKey` action to determine the `key Id` of the EBS-managed key in your account.
- Fetch the snapshot of the EBS volumes encrypted using AWS managed key, from your AWS account and copy it to the [GuardDuty service account]. For this purpose, we use the permissions

GetSnapshotBlock and ListSnapshotBlocks. GuardDuty will then scan the snapshot in the service account. Presently, the Malware Protection for EC2 support for scanning EBS volumes encrypted with AWS managed key might not be available in all the AWS Regions. For more information, see Region-specific feature availability.

- Allow Amazon EC2 to call AWS KMS on behalf of Malware Protection for EC2 to perform several cryptographic actions on customer managed keys. Actions such as `kms:ReEncryptTo` and `kms:ReEncryptFrom` are required to share the snapshots that are encrypted with the customer managed keys. Only those keys are accessible for which the `GuardDutyExcluded` tag is not set to `true`.

The role is configured with the following AWS managed policy, named `AmazonGuardDutyMalwareProtectionServiceRolePolicy`.

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [{
            "Sid": "DescribeAndListPermissions",
            "Effect": "Allow",
            "Action": [
                "ec2:DescribeInstances",
                "ec2:DescribeVolumes",
                "ec2:DescribeSnapshots",
                "ecs:ListClusters",
                "ecs:ListContainerInstances",
                "ecs:ListTasks",
                "ecs:DescribeTasks",
                "eks:DescribeCluster"
            ],
            "Resource": "*"
        },
        {
            "Sid": "CreateSnapshotVolumeConditionalStatement",
            "Effect": "Allow",
            "Action": "ec2:CreateSnapshot",
            "Resource": "arn:aws:ec2:*:*:volume/*",
            "Condition": {
                "Null": {
                    "aws:ResourceTag/GuardDutyExcluded": "true"
```

```
                }
            }
        },
        {
            "Sid": "CreateSnapshotConditionalStatement",
            "Effect": "Allow",
            "Action": "ec2:CreateSnapshot",
            "Resource": "arn:aws:ec2:*:*:snapshot/*",
            "Condition": {
                "ForAnyValue:StringEquals": {
                    "aws:TagKeys": "GuardDutyScanId"
                }
            }
        },
        {
            "Sid": "CreateTagsPermission",
            "Effect": "Allow",
            "Action": "ec2:CreateTags",
            "Resource": "arn:aws:ec2:*:*:*/*",
            "Condition": {
                "StringEquals": {
                    "ec2:CreateAction": "CreateSnapshot"
                }
            }
        },
        {
            "Sid": "AddTagsToSnapshotPermission",
            "Effect": "Allow",
            "Action": "ec2:CreateTags",
            "Resource": "arn:aws:ec2:*:*:snapshot/*",
            "Condition": {
                "StringLike": {
                    "ec2:ResourceTag/GuardDutyScanId": "*"
                },
                "ForAllValues:StringEquals": {
                    "aws:TagKeys": [
                        "GuardDutyExcluded",
                        "GuardDutyFindingDetected"
                    ]
                }
            }
        },
        {
            "Sid": "DeleteAndShareSnapshotPermission",
```

```
                "Effect": "Allow",
                "Action": [
                    "ec2:DeleteSnapshot",
                    "ec2:ModifySnapshotAttribute"
                ],
                "Resource": "arn:aws:ec2:*:*:snapshot/*",
                "Condition": {
                    "StringLike": {
                        "ec2:ResourceTag/GuardDutyScanId": "*"
                    },
                    "Null": {
                        "aws:ResourceTag/GuardDutyExcluded": "true"
                    }
                }
            },
            {
                "Sid": "PreventPublicAccessToSnapshotPermission",
                "Effect": "Deny",
                "Action": [
                    "ec2:ModifySnapshotAttribute"
                ],
                "Resource": "arn:aws:ec2:*:*:snapshot/*",
                "Condition": {
                    "StringEquals": {
                        "ec2:Add/group": "all"
                    }
                }
            },
            {
                "Sid": "CreateGrantPermission",
                "Effect": "Allow",
                "Action": "kms:CreateGrant",
                "Resource": "arn:aws:kms:*:*:key/*",
                "Condition": {
                    "Null": {
                        "aws:ResourceTag/GuardDutyExcluded": "true"
                    },
                    "StringLike": {
                        "kms:EncryptionContext:aws:ebs:id": "snap-*"
                    },
                    "ForAllValues:StringEquals": {
                        "kms:GrantOperations": [
                            "Decrypt",
                            "CreateGrant",
```

```
                        "GenerateDataKeyWithoutPlaintext",
                        "ReEncryptFrom",
                        "ReEncryptTo",
                        "RetireGrant",
                        "DescribeKey"
                    ]
                },
                "Bool": {
                    "kms:GrantIsForAWSResource": "true"
                }
            }
        },
        {
            "Sid": "ShareSnapshotKMSPermission",
            "Effect": "Allow",
            "Action": [
                "kms:ReEncryptTo",
                "kms:ReEncryptFrom"
            ],
            "Resource": "arn:aws:kms:*:*:key/*",
            "Condition": {
                "StringLike": {
                    "kms:ViaService": "ec2.*.amazonaws.com"
                },
                "Null": {
                    "aws:ResourceTag/GuardDutyExcluded": "true"
                }
            }
        },
        {
            "Sid": "DescribeKeyPermission",
            "Effect": "Allow",
            "Action": "kms:DescribeKey",
            "Resource": "arn:aws:kms:*:*:key/*"
        },
        {
            "Sid": "GuardDutyLogGroupPermission",
            "Effect": "Allow",
            "Action": [
                "logs:DescribeLogGroups",
                "logs:CreateLogGroup",
                "logs:PutRetentionPolicy"
            ],
            "Resource": "arn:aws:logs:*:*:log-group:/aws/guardduty/*"
```

```
        },
        {
            "Sid": "GuardDutyLogStreamPermission",
            "Effect": "Allow",
            "Action": [
                "logs:CreateLogStream",
                "logs:PutLogEvents",
                "logs:DescribeLogStreams"
            ],
            "Resource": "arn:aws:logs:*:*:log-group:/aws/guardduty/*:log-
stream:*"
        },
        {
            "Sid": "EBSDirectAPIPermissions",
            "Effect": "Allow",
            "Action": [
                "ebs:GetSnapshotBlock",
                "ebs:ListSnapshotBlocks"
            ],
            "Resource": "arn:aws:ec2:*:*:snapshot/*",
            "Condition": {
                "StringLike": {
                    "aws:ResourceTag/GuardDutyScanId": "*"
                },
                "Null": {
                    "aws:ResourceTag/GuardDutyExcluded": "true"
                }
            }
        }
    ]
}
```

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [{
            "Sid": "DescribeAndListPermissions",
            "Effect": "Allow",
            "Action": [
                "ec2:DescribeInstances",
                "ec2:DescribeVolumes",
```

```
                "ec2:DescribeSnapshots",
                "ecs:ListClusters",
                "ecs:ListContainerInstances",
                "ecs:ListTasks",
                "ecs:DescribeTasks",
                "eks:DescribeCluster"
            ],
            "Resource": "*"
        },
        {
            "Sid": "CreateSnapshotVolumeConditionalStatement",
            "Effect": "Allow",
            "Action": "ec2:CreateSnapshot",
            "Resource": "arn:aws:ec2:*:*:volume/*",
            "Condition": {
                "Null": {
                    "aws:ResourceTag/GuardDutyExcluded": "true"
                }
            }
        },
        {
            "Sid": "CreateSnapshotConditionalStatement",
            "Effect": "Allow",
            "Action": "ec2:CreateSnapshot",
            "Resource": "arn:aws:ec2:*:*:snapshot/*",
            "Condition": {
                "ForAnyValue:StringEquals": {
                    "aws:TagKeys": "GuardDutyScanId"
                }
            }
        },
        {
            "Sid": "CreateTagsPermission",
            "Effect": "Allow",
            "Action": "ec2:CreateTags",
            "Resource": "arn:aws:ec2:*:*:*/*",
            "Condition": {
                "StringEquals": {
                    "ec2:CreateAction": "CreateSnapshot"
                }
            }
        },
        {
            "Sid": "AddTagsToSnapshotPermission",
```

```
            "Effect": "Allow",
            "Action": "ec2:CreateTags",
            "Resource": "arn:aws:ec2:*:*:snapshot/*",
            "Condition": {
                "StringLike": {
                    "ec2:ResourceTag/GuardDutyScanId": "*"
                },
                "ForAllValues:StringEquals": {
                    "aws:TagKeys": [
                        "GuardDutyExcluded",
                        "GuardDutyFindingDetected"
                    ]
                }
            }
        },
        {
            "Sid": "DeleteAndShareSnapshotPermission",
            "Effect": "Allow",
            "Action": [
                "ec2:DeleteSnapshot",
                "ec2:ModifySnapshotAttribute"
            ],
            "Resource": "arn:aws:ec2:*:*:snapshot/*",
            "Condition": {
                "StringLike": {
                    "ec2:ResourceTag/GuardDutyScanId": "*"
                },
                "Null": {
                    "aws:ResourceTag/GuardDutyExcluded": "true"
                }
            }
        },
        {
            "Sid": "PreventPublicAccessToSnapshotPermission",
            "Effect": "Deny",
            "Action": [
                "ec2:ModifySnapshotAttribute"
            ],
            "Resource": "arn:aws:ec2:*:*:snapshot/*",
            "Condition": {
                "StringEquals": {
                    "ec2:Add/group": "all"
                }
            }
```

```
        },
        {
            "Sid": "CreateGrantPermission",
            "Effect": "Allow",
            "Action": "kms:CreateGrant",
            "Resource": "arn:aws:kms:*:*:key/*",
            "Condition": {
                "Null": {
                    "aws:ResourceTag/GuardDutyExcluded": "true"
                },
                "StringLike": {
                    "kms:EncryptionContext:aws:ebs:id": "snap-*"
                },
                "ForAllValues:StringEquals": {
                    "kms:GrantOperations": [
                        "Decrypt",
                        "CreateGrant",
                        "GenerateDataKeyWithoutPlaintext",
                        "ReEncryptFrom",
                        "ReEncryptTo",
                        "RetireGrant",
                        "DescribeKey"
                    ]
                },
                "Bool": {
                    "kms:GrantIsForAWSResource": "true"
                }
            }
        },
        {
            "Sid": "ShareSnapshotKMSPermission",
            "Effect": "Allow",
            "Action": [
                "kms:ReEncryptTo",
                "kms:ReEncryptFrom"
            ],
            "Resource": "arn:aws:kms:*:*:key/*",
            "Condition": {
                "StringLike": {
                    "kms:ViaService": "ec2.*.amazonaws.com"
                },
                "Null": {
                    "aws:ResourceTag/GuardDutyExcluded": "true"
                }
```

```
            }
        },
        {
            "Sid": "DescribeKeyPermission",
            "Effect": "Allow",
            "Action": "kms:DescribeKey",
            "Resource": "arn:aws:kms:*:*:key/*"
        },
        {
            "Sid": "GuardDutyLogGroupPermission",
            "Effect": "Allow",
            "Action": [
                "logs:DescribeLogGroups",
                "logs:CreateLogGroup",
                "logs:PutRetentionPolicy"
            ],
            "Resource": "arn:aws:logs:*:*:log-group:/aws/guardduty/*"
        },
        {
            "Sid": "GuardDutyLogStreamPermission",
            "Effect": "Allow",
            "Action": [
                "logs:CreateLogStream",
                "logs:PutLogEvents",
                "logs:DescribeLogStreams"
            ],
            "Resource": "arn:aws:logs:*:*:log-group:/aws/guardduty/*:log-
stream:*"
        }
    ]
}
```

The following trust policy is attached to the `AWSServiceRoleForAmazonGuardDutyMalwareProtection` service-linked role:

JSON

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
```

```
        "Effect": "Allow",
        "Principal": {
          "Service": "malware-protection.guardduty.amazonaws.com"
        },
        "Action": "sts:AssumeRole"
      }
    ]
}
```

Creating a service-linked role for Malware Protection for EC2

The `AWSServiceRoleForAmazonGuardDutyMalwareProtection` service-linked role is automatically created when you enable Malware Protection for EC2 for the first time or enable Malware Protection for EC2 in a supported Region where you previously didn't have it enabled. You can also create the `AWSServiceRoleForAmazonGuardDutyMalwareProtection` service-linked role manually using the IAM console, the IAM CLI, or the IAM API.

> **Note**
>
> By default, if you are new to Amazon GuardDuty, Malware Protection for EC2 is automatically enabled.

> **Important**
>
> The service-linked role that is created for the delegated GuardDuty administrator account doesn't apply to the member GuardDuty accounts.

You must configure permissions to allow an IAM principal (such as a user, group, or role) to create, edit, or delete a service-linked role. For the `AWSServiceRoleForAmazonGuardDutyMalwareProtection` service-linked role to be successfully created, the IAM identity that you use GuardDuty with must have the required permissions. To grant the required permissions, attach the following policy to this user, group, or role:

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [{
            "Effect": "Allow",
            "Action": "guardduty:*",
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": "iam:CreateServiceLinkedRole",
            "Resource": "*",
            "Condition": {
                "StringLike": {
                    "iam:AWSServiceName": [
                        "malware-protection.guardduty.amazonaws.com"
                    ]
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": [
                "organizations:EnableAWSServiceAccess",
                "organizations:RegisterDelegatedAdministrator",
                "organizations:ListDelegatedAdministrators",
                "organizations:ListAWSServiceAccessForOrganization",
                "organizations:DescribeOrganizationalUnit",
                "organizations:DescribeAccount",
                "organizations:DescribeOrganization"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": "iam:GetRole",
            "Resource": "arn:aws:iam::*:role/
*AWSServiceRoleForAmazonGuardDutyMalwareProtection"
        }
    ]
}
```

For more information about creating the role manually, see Creating a service-linked role in the *IAM User Guide*.

Editing a service-linked role for Malware Protection for EC2

Malware Protection for EC2 doesn't allow you to edit the `AWSServiceRoleForAmazonGuardDutyMalwareProtection` service-linked role. After you create a service-linked role, you can't change the name of the role because various entities might reference the role. However, you can edit the description of the role using IAM. For more information, see Editing a service-linked role in the *IAM User Guide*.

Deleting a service-linked role for Malware Protection for EC2

If you no longer need to use a feature or service that requires a service-linked role, we recommend that you delete that role. That way you don't have an unused entity that isn't actively monitored or maintained.

> **Important**
>
> In order to delete the `AWSServiceRoleForAmazonGuardDutyMalwareProtection`, you must first disable Malware Protection for EC2 in all of the Regions where it is enabled. If Malware Protection for EC2 isn't disabled when you try to delete the service-linked role, the deletion will fail. Ensure that you first disable Malware Protection for EC2 in your account.

When you choose **Disable** to stop the Malware Protection for EC2 service, the `AWSServiceRoleForAmazonGuardDutyMalwareProtection` is not automatically deleted. If you then choose **Enable** to start the Malware Protection for EC2 service again, GuardDuty will start using the existing `AWSServiceRoleForAmazonGuardDutyMalwareProtection`.

To manually delete the service-linked role using IAM

Use the IAM console, the AWS CLI, or the IAM API to delete the `AWSServiceRoleForAmazonGuardDutyMalwareProtection` service-linked role. For more information, see Deleting a service-linked role in the *IAM User Guide*.

Supported AWS Regions

Amazon GuardDuty supports using the `AWSServiceRoleForAmazonGuardDutyMalwareProtection` service-linked role in all the AWS Regions where Malware Protection for EC2 is available.

For a list of Regions where GuardDuty is currently available, see Amazon GuardDuty endpoints and quotas in the *Amazon Web Services General Reference*.

> **Note**
>
> Malware Protection for EC2 is currently unavailable in AWS GovCloud (US-East) and AWS GovCloud (US-West).

AWS managed policies for Amazon GuardDuty

To add permissions to users, groups, and roles, it is easier to use AWS managed policies than to write policies yourself. It takes time and expertise to create IAM customer managed policies that provide your team with only the permissions they need. To get started quickly, you can use our AWS managed policies. These policies cover common use cases and are available in your AWS account. For more information about AWS managed policies, see AWS managed policies in the *IAM User Guide*.

AWS services maintain and update AWS managed policies. You can't change the permissions in AWS managed policies. Services occasionally add additional permissions to an AWS managed policy to support new features. This type of update affects all identities (users, groups, and roles) where the policy is attached. Services are most likely to update an AWS managed policy when a new feature is launched or when new operations become available. Services do not remove permissions from an AWS managed policy, so policy updates won't break your existing permissions.

Additionally, AWS supports managed policies for job functions that span multiple services. For example, the **ReadOnlyAccess** AWS managed policy provides read-only access to all AWS services and resources. When a service launches a new feature, AWS adds read-only permissions for new operations and resources. For a list and descriptions of job function policies, see AWS managed policies for job functions in the *IAM User Guide*.

The `Version` policy element specifies the language syntax rules that are to be used to process a policy. The following policies include the current version that IAM supports. For more information, see IAM JSON policy elements: Version.

AWS managed policy: AmazonGuardDutyFullAccess_v2 (recommended)

You can attach the AmazonGuardDutyFullAccess_v2 policy to your IAM identities. This policy will allow a user full access to perform all GuardDuty actions and access required resources. Between AmazonGuardDutyFullAccess_v2 and AmazonGuardDutyFullAccess, GuardDuty recommends attaching AmazonGuardDutyFullAccess_v2 because it offers enhanced security and restricts administrative actions to GuardDuty service principals.

Permission details

The AmazonGuardDutyFullAccess_v2 policy includes the following permissions:

- `GuardDuty` – Allows users full access to all GuardDuty actions.
- `IAM`:
  - Allows users to create GuardDuty service-linked role.
  - Allows viewing and managing IAM roles and their policies for GuardDuty.
  - Allows users to pass a role to GuardDuty that uses this role to enable the GuardDuty Malware Protection for S3 feature. This is regardless of how you enable Malware Protection for S3 - within the GuardDuty service or independently.
  - The permission to perform an `iam:GetRole` action on `AWSServiceRoleForAmazonGuardDutyMalwareProtection` establishes if the service-linked role (SLR) for Malware Protection for EC2 exists in an account.
- `Organizations`:
  - Allow users to read (view) GuardDuty organization structure and accounts.
  - Allows users to designate a delegated administrator and manage members for a GuardDuty organization.

To review the permissions for this policy, see AmazonGuardDutyFullAccess_v2 in the *AWS Managed Policy Reference Guide*.

AWS managed policy: AmazonGuardDutyFullAccess

You can attach the `AmazonGuardDutyFullAccess` policy to your IAM identities.

> **Important**
>
> For enhanced security and restrictive permissions to GuardDuty service principals, we recommend you to use AWS managed policy: AmazonGuardDutyFullAccess_v2 (recommended).

This policy grants administrative permissions that allow a user full access to perform all GuardDuty actions and resources.

Permission details

This policy includes the following permissions.

- `GuardDuty` – Allows users full access to all GuardDuty actions.
- `IAM`:
  - Allows users to create the GuardDuty service-linked role.
  - Allows an administrator account to enable GuardDuty for member accounts.
  - Allows users to pass a role to GuardDuty that uses this role to enable the GuardDuty Malware Protection for S3 feature. This is regardless of how you enable Malware Protection for S3 - within the GuardDuty service or independently.
- `Organizations` – Allows users to designate a delegated administrator and manage members for a GuardDuty organization.

The permission to perform an `iam:GetRole` action on `AWSServiceRoleForAmazonGuardDutyMalwareProtection` establishes if the service-linked role (SLR) for Malware Protection for EC2 exists in an account.

To review the permissions for this policy, see AmazonGuardDutyFullAccess in the *AWS Managed Policy Reference Guide*.

AWS managed policy: AmazonGuardDutyReadOnlyAccess

You can attach the `AmazonGuardDutyReadOnlyAccess` policy to your IAM identities.

This policy grants read-only permissions that allow a user to view GuardDuty findings and details of your GuardDuty organization.

Permissions details

This policy includes the following permissions.

- `GuardDuty` – Allows users to view GuardDuty findings and perform API operations that start with `Get`, `List`, or `Describe`.
- `Organizations` – Allows users to retrieve information about your GuardDuty organization configuration, including details of the delegated administrator account.

To review the permissions for this policy, see AmazonGuardDutyReadOnlyAccess in the *AWS Managed Policy Reference Guide*.

AWS managed policy: AmazonGuardDutyServiceRolePolicy

You can't attach `AmazonGuardDutyServiceRolePolicy` to your IAM entities. This AWS managed policy is attached to a service-linked role that allows GuardDuty to perform actions on your behalf. For more information, see Service-linked role permissions for GuardDuty.

GuardDuty updates to AWS managed policies

View details about updates to AWS managed policies for GuardDuty since this service began tracking these changes. For automatic alerts about changes to this page, subscribe to the RSS feed on the GuardDuty Document history page.

Change	Description	Date
AmazonGuardDutyFullAccess_v2 – Added a new policy	Added a new AmazonGuardDutyFullAccess_v2 policy. This is recommended because its permissions enhance security by restricting administrative actions to GuardDuty service principals based on IAM roles and	June 04, 2025

<table>
<tr><th>Change</th><th>Description</th><th>Date</th></tr>
<tr><td></td><td>policies, and AWS Organizations integration.</td><td></td></tr>
<tr><td>AmazonGuardDutyServiceRolePolicy – Update to an existing policy</td><td>Added the <code>ec2:DescribeVpcs</code> permission. This allows GuardDuty to track VPC updates, such as retrieving the VPC CIDR.</td><td>August 22, 2024</td></tr>
<tr><td>AmazonGuardDutyServiceRolePolicy – Update to an existing policy</td><td>Added permission that allows you to pass an IAM role to GuardDuty when you enable Malware Protection for S3.
<pre>{
            "Sid":
 "AllowPassRoleToMa
lwareProtectionPlan",
            "Effect":
 "Allow",
            "Action": [

 "iam:PassRole"
            ],
            "Resource":
 "arn:aws:iam::*:role/
*",
            "Conditio
n": {

 "StringEquals": {

 "iam:PassedToServi
ce": "guarddut
y.amazonaws.com"
                }
            }
}</pre></td><td>June 10, 2024</td></tr>
</table>

Change	Description	Date
AmazonGuardDutyServiceRolePolicy – Update to an existing policy.	Use AWS Systems Manager actions to manage SSM associations on Amazon EC2 instances when you enable GuardDuty Runtime Monitoring with automated agent for Amazon EC2. When GuardDuty automated agent configuration is disabled, GuardDuty considers only those EC2 instances that have an inclusion tag (`GuardDuty Managed :true`).	March 26, 2024
AmazonGuardDutyServiceRolePolicy – Update to an existing policy.	GuardDuty has added a new permission - `organization:DescribeOrganization` to retrieve the organization ID of the shared Amazon VPC account and set the Amazon VPC endpoint policy with organization ID.	February 9, 2024
AmazonGuardDutyMalwareProtectionServiceRolePolicy – Update to an existing policy.	Malware Protection for EC2 has added two permissions - `GetSnapshotBlock` and `ListSnapshotBlocks` to fetch the snapshot of an EBS volume (encrypted using AWS managed key) from your AWS account and copy it to the GuardDuty service account before starting the malware scan.	Jan 25, 2024

Change	Description	Date
AmazonGuardDutyServiceRolePolicy – Update to an existing policy	Added new permissions to allow GuardDuty to add `guarddutyActivate` Amazon ECS account setting, and perform list and describe operations on Amazon ECS clusters.	Nov 26, 2023
AmazonGuardDutyReadOnlyAccess – Update to an existing policy	GuardDuty added a new policy for `organizations` to `ListAccounts` .	November 16, 2023
AmazonGuardDutyFullAccess – Update to an existing policy	GuardDuty added a new policy for `organizations` to `ListAccounts` .	November 16, 2023
AmazonGuardDutyServiceRolePolicy – Update to an existing policy	GuardDuty added new permissions to support the upcoming GuardDuty EKS Runtime Monitoring feature.	March 8, 2023

Change	Description	Date
AmazonGuardDutyServiceRolePolicy – Update to an existing policy	GuardDuty has added new permissions to allow GuardDuty to create Service-linked role for Malware Protection for EC2. This will help GuardDuty streamline the process of enabling Malware Protection for EC2. GuardDuty can now perform the following IAM action: `{ "Effect": "Allow", "Action": "iam:CreateServiceLinkedRole", "Resource": "*", "Condition": { "StringEquals": { "iam:AWSServiceName": "malware-protection.guardduty.amazonaws.com" } } }`	Feb 21, 2023
AmazonGuardDutyFullAccess – Update to an existing policy	GuardDuty updated ARN for `iam:GetRole` to `*AWSServiceRoleForAmazonGuardDutyMalwareProtection` .	Jul 26, 2022

Change	Description	Date
AmazonGuardDutyFullAccess – Update to an existing policy	GuardDuty added a new `AWSServiceName` to allow the creation of service-linked role using `iam:CreateServiceLinkedRole` for GuardDuty Malware Protection for EC2 service. GuardDuty can now perform the `iam:GetRole` action to gain information for `AWSServiceRole` .	Jul 26, 2022
AmazonGuardDutyServiceRolePolicy – Update to an existing policy	GuardDuty added new permissions to allow GuardDuty to use Amazon EC2 networking actions to improve findings. GuardDuty can now perform the following EC2 actions to gain information about how your EC2 instances are communicating. This information is used to improve finding accuracy. • `ec2:DescribeVpcEndpoints` • `ec2:DescribeSubnets` • `ec2:DescribeVpcPeeringConnections` • `ec2:DescribeTransitGatewayAttachments`	Aug 3, 2021

Change	Description	Date
GuardDuty started tracking changes	GuardDuty started tracking changes for its AWS managed policies.	Aug 3, 2021

Troubleshooting Amazon GuardDuty identity and access

Use the following information to help you diagnose and fix common issues that you might encounter when working with GuardDuty and IAM.

Topics

- I am not authorized to perform an action in GuardDuty
- I'm not authorized to perform iam:PassRole.
- I want to allow people outside of my AWS account to access my GuardDuty resources.

I am not authorized to perform an action in GuardDuty

If you receive an error that you're not authorized to perform an action, your policies must be updated to allow you to perform the action.

The following example error occurs when the `mateojackson` IAM user tries to use the console to view details about a fictional *`my-example-widget`* resource but doesn't have the fictional `guardduty:`*`GetWidget`* permissions.

```
User: arn:aws:iam::123456789012:user/mateojackson is not authorized to perform:
 guardduty:GetWidget on resource: my-example-widget
```

In this case, the policy for the `mateojackson` user must be updated to allow access to the *`my-example-widget`* resource by using the `guardduty:`*`GetWidget`* action.

If you need help, contact your AWS administrator. Your administrator is the person who provided you with your sign-in credentials.

I'm not authorized to perform iam:PassRole.

If you receive an error that you're not authorized to perform the `iam:PassRole` action, your policies must be updated to allow you to pass a role to GuardDuty.

Some AWS services allow you to pass an existing role to that service instead of creating a new service role or service-linked role. To do this, you must have permissions to pass the role to the service.

The following example error occurs when an IAM user named `marymajor` tries to use the console to perform an action in GuardDuty. However, the action requires the service to have permissions that are granted by a service role. Mary does not have permissions to pass the role to the service.

```
User: arn:aws:iam::123456789012:user/marymajor is not authorized to perform:
 iam:PassRole
```

In this case, Mary's policies must be updated to allow her to perform the `iam:PassRole` action.

If you need help, contact your AWS administrator. Your administrator is the person who provided you with your sign-in credentials.

I want to allow people outside of my AWS account to access my GuardDuty resources.

You can create a role that users in other accounts or people outside of your organization can use to access your resources. You can specify who is trusted to assume the role. For services that support resource-based policies or access control lists (ACLs), you can use those policies to grant people access to your resources.

To learn more, consult the following:

- To learn whether GuardDuty supports these features, see How Amazon GuardDuty works with IAM.
- To learn how to provide access to your resources across AWS accounts that you own, see Providing access to an IAM user in another AWS account that you own in the *IAM User Guide*.
- To learn how to provide access to your resources to third-party AWS accounts, see Providing access to AWS accounts owned by third parties in the *IAM User Guide*.
- To learn how to provide access through identity federation, see Providing access to externally authenticated users (identity federation) in the *IAM User Guide*.
- To learn the difference between using roles and resource-based policies for cross-account access, see Cross account resource access in IAM in the *IAM User Guide*.

Compliance validation for Amazon GuardDuty

To learn whether an AWS service is within the scope of specific compliance programs, see AWS services in Scope by Compliance Program and choose the compliance program that you are interested in. For general information, see AWS Compliance Programs.

You can download third-party audit reports using AWS Artifact. For more information, see Downloading Reports in AWS Artifact.

Your compliance responsibility when using AWS services is determined by the sensitivity of your data, your company's compliance objectives, and applicable laws and regulations. AWS provides the following resources to help with compliance:

- Security Compliance & Governance – These solution implementation guides discuss architectural considerations and provide steps for deploying security and compliance features.
- HIPAA Eligible Services Reference – Lists HIPAA eligible services. Not all AWS services are HIPAA eligible.
- AWS Compliance Resources – This collection of workbooks and guides might apply to your industry and location.
- AWS Customer Compliance Guides – Understand the shared responsibility model through the lens of compliance. The guides summarize the best practices for securing AWS services and map the guidance to security controls across multiple frameworks (including National Institute of Standards and Technology (NIST), Payment Card Industry Security Standards Council (PCI), and International Organization for Standardization (ISO)).
- Evaluating Resources with Rules in the *AWS Config Developer Guide* – The AWS Config service assesses how well your resource configurations comply with internal practices, industry guidelines, and regulations.
- AWS Security Hub – This AWS service provides a comprehensive view of your security state within AWS. Security Hub uses security controls to evaluate your AWS resources and to check your compliance against security industry standards and best practices. For a list of supported services and controls, see Security Hub controls reference.
- Amazon GuardDuty – This AWS service detects potential threats to your AWS accounts, workloads, containers, and data by monitoring your environment for suspicious and malicious activities. GuardDuty can help you address various compliance requirements, like PCI DSS, by meeting intrusion detection requirements mandated by certain compliance frameworks.

- AWS Audit Manager – This AWS service helps you continuously audit your AWS usage to simplify how you manage risk and compliance with regulations and industry standards.

Resilience in Amazon GuardDuty

The AWS global infrastructure is built around AWS Regions and Availability Zones. Regions provide multiple physically separated and isolated Availability Zones, which are connected through low-latency, high-throughput, and highly redundant networking. With Availability Zones, you can design and operate applications and databases that automatically fail over between zones without interruption. Availability Zones are more highly available, fault tolerant, and scalable than traditional single or multiple data center infrastructures.

For more information about AWS Regions and Availability Zones, see AWS global infrastructure.

Infrastructure security in Amazon GuardDuty

As a managed service, Amazon GuardDuty is protected by AWS global network security. For information about AWS security services and how AWS protects infrastructure, see AWS Cloud Security. To design your AWS environment using the best practices for infrastructure security, see Infrastructure Protection in *Security Pillar AWS Well‐Architected Framework*.

You use AWS published API calls to access GuardDuty through the network. Clients must support the following:

- Transport Layer Security (TLS). We require TLS 1.2 and recommend TLS 1.3.
- Cipher suites with perfect forward secrecy (PFS) such as DHE (Ephemeral Diffie-Hellman) or ECDHE (Elliptic Curve Ephemeral Diffie-Hellman). Most modern systems such as Java 7 and later support these modes.

Additionally, requests must be signed by using an access key ID and a secret access key that is associated with an IAM principal. Or you can use the AWS Security Token Service (AWS STS) to generate temporary security credentials to sign requests.

Amazon GuardDuty and interface VPC endpoints (AWS PrivateLink)

You can establish a private connection between your VPC and Amazon GuardDuty by creating an *interface VPC endpoint*. Interface endpoints are powered by AWS PrivateLink, a technology that enables you to privately access GuardDuty APIs without an internet gateway, NAT device, VPN connection, or AWS Direct Connect connection. Instances in your VPC don't need public IP addresses to communicate with GuardDuty APIs. Traffic between your VPC and GuardDuty does not leave the Amazon network.

Each interface endpoint is represented by one or more Elastic Network Interfaces in your subnets.

For more information, see Interface VPC endpoints (AWS PrivateLink) in the *AWS PrivateLink Guide*.

Considerations for GuardDuty VPC endpoints

Before you set up an interface VPC endpoint for GuardDuty, ensure that you review Interface endpoint properties and limitations in the *AWS PrivateLink Guide*.

GuardDuty supports making calls to all of its API actions from your VPC.

Creating an interface VPC endpoint for GuardDuty

You can create a VPC endpoint for the GuardDuty service using either the Amazon VPC console or the AWS Command Line Interface (AWS CLI). For more information, see Create an interface endpoint in the *AWS PrivateLink Guide*.

Create a VPC endpoint for GuardDuty using the following service name:

- com.amazonaws.*region*.guardduty
- com.amazonaws.*region*.guardduty-fips (FIPS endpoint)

If you enable private DNS for the endpoint, you can make API requests to GuardDuty using its default DNS name for the Region, for example, `guardduty.us-east-1.amazonaws.com`.

For more information, see Access a service through an interface endpoint in the *AWS PrivateLink Guide*.

Creating a VPC endpoint policy for GuardDuty

You can attach an endpoint policy to your VPC endpoint that controls access to GuardDuty. The policy specifies the following information:

- The principal that can perform actions.
- The actions that can be performed.
- The resources on which actions can be performed.

For more information, see Control access to services with VPC endpoints in the *AWS PrivateLink Guide*.

Example: VPC endpoint policy for GuardDuty actions

The following is an example of an endpoint policy for GuardDuty. When attached to an endpoint, this policy grants access to the listed GuardDuty actions for all principals on all resources.

```
{
   "Statement":[
      {
         "Principal":"*",
         "Effect":"Allow",
         "Action":[
            "guardduty:listDetectors",
            "guardduty:getDetector",
            "guardduty:getFindings"
         ],
         "Resource":"*"
      }
   ]
}
```

Shared subnets

You can't create, describe, modify, or delete VPC endpoints in subnets that are shared with you. However, you can use the VPC endpoints in subnets that are shared with you. For information about VPC sharing, see Share your VPC with other accounts in the *Amazon VPC User Guide*.

GuardDuty integrating with AWS security services

GuardDuty can be integrated with other AWS security services. These services can ingest data from GuardDuty to allow you to view findings in new ways. Review the following integration options to learn more about how that service is set up to work with GuardDuty.

Integrating GuardDuty with AWS Security Hub

AWS Security Hub collects security data from across your AWS accounts, services, and supported third party partner products to assess the security state of your environment according to industry standards and best practices. In addition to evaluating your security posture, Security Hub creates a central location for findings across all of your integrated AWS services, and AWS Partner products. Enabling Security Hub with GuardDuty will automatically allow GuardDuty findings data to be ingested by Security Hub.

For more information about using Security Hub with GuardDuty see Integrating with AWS Security Hub.

Integrating GuardDuty with Amazon Detective

Amazon Detective uses log data from across your AWS accounts to create data visualizations for your resources and IP addresses interacting with your environment. Detective's visualizations help you quickly and easily investigate security issues. You can pivot from GuardDuty finding details to information in the Detective console once both services are enabled.

For more information about using Detective with GuardDuty see Integrating with Amazon Detective.

Integrating with AWS Security Hub

AWS Security Hub provides you with a comprehensive view of your security state in AWS and helps you to check your environment against security industry standards and best practices. Security Hub collects security data from across AWS accounts, services, and supported third-party partner products and helps you to analyze your security trends and identify the highest priority security issues.

The Amazon GuardDuty integration with Security Hub enables you to send findings from GuardDuty to Security Hub. Security Hub can then include those findings in its analysis of your security posture.

Contents

- How Amazon GuardDuty sends findings to AWS Security Hub
  - Types of findings that GuardDuty sends to Security Hub
    - Latency for sending new findings
    - Retrying when Security Hub is not available
    - Updating existing findings in Security Hub
- Viewing GuardDuty findings in AWS Security Hub
  - Interpreting GuardDuty finding names in AWS Security Hub
  - Typical finding from GuardDuty
- Enabling and configuring the integration
- Using GuardDuty controls in Security Hub
- Stopping the publication of findings to Security Hub

How Amazon GuardDuty sends findings to AWS Security Hub

In AWS Security Hub, security issues are tracked as findings. Some findings come from issues that are detected by other AWS services or by third-party partners. Security Hub also has a set of rules that it uses to detect security issues and generate findings.

Security Hub provides tools to manage findings from across all of these sources. You can view and filter lists of findings and view details for a finding. For more information, see Viewing findings in the *AWS Security Hub User Guide*. You can also track the status of an investigation into a finding. For more information, see Taking action on findings in the *AWS Security Hub User Guide*.

All findings in Security Hub use a standard JSON format called the AWS Security Finding Format (ASFF). The ASFF includes details about the source of the issue, the affected resources, and the current status of the finding. See AWS Security Finding Format (ASFF) in the *AWS Security Hub User Guide*.

Amazon GuardDuty is one of the AWS services that sends findings to Security Hub.

Types of findings that GuardDuty sends to Security Hub

Once you enable GuardDuty and Security Hub in the same account within the same AWS Region, GuardDuty starts sending all the generated findings to Security Hub. These findings are sent to Security Hub using the AWS Security Finding Format (ASFF). In ASFF, the `Types` field provides the finding type.

Latency for sending new findings

When GuardDuty creates a new finding, it is usually sent to Security Hub within five minutes.

Retrying when Security Hub is not available

If Security Hub is not available, GuardDuty retries sending the findings until they are received.

Updating existing findings in Security Hub

After it sends a finding to Security Hub, GuardDuty sends updates to reflect additional observations of the finding activity to Security Hub. The new observations of these findings are sent to Security Hub based on the Step 5 – Frequency for exporting findings settings in your AWS account.

When you archive or unarchive a finding, GuardDuty doesn't send that finding to Security Hub. Any manually unarchived finding that later become active in GuardDuty is not sent to Security Hub.

Viewing GuardDuty findings in AWS Security Hub

Sign in to the AWS Management Console and open the AWS Security Hub console at https://console.aws.amazon.com/securityhub/.

You can now use either of the following ways to view the GuardDuty findings in the Security Hub console:

Option 1: Using *Integrations* in Security Hub

1. In the left navigation pane, choose **Integrations**.
2. On the **Integrations** page, check the **Status** for **Amazon: GuardDuty**.
   - If the **Status** is **Accepting findings**, then choose **See findings** next to **Accepting findings**.
   - If not, then for more information about how **Integrations** work, see Security Hub integrations in *AWS Security Hub User Guide*.

Option 2: Using *Findings* in Security Hub

1. In the left navigation pane, choose **Findings**.
2. On the **Findings** page, add the filter **Product name** and enter **`GuardDuty`** to view only GuardDuty findings.

Interpreting GuardDuty finding names in AWS Security Hub

GuardDuty sends the findings to Security Hub using the AWS Security Finding Format (ASFF). In ASFF, the `Types` field provides the finding type. ASFF types use a different naming scheme than GuardDuty types. The table below details all the GuardDuty finding types with their ASFF counterpart as they appear in Security Hub.

> **Note**
>
> For some GuardDuty finding types Security Hub assigns different ASFF finding names depending on whether the finding detail's **Resource Role** was **ACTOR** or **TARGET**. For more information see Finding details.

GuardDuty finding type	ASFF finding type
AttackSequence:IAM/CompromisedCredentials	TTPs/AttackSequence:IAM/CompromisedCredentials
AttackSequence:S3/CompromisedData	TTPs/AttackSequence:S3/CompromisedData
Backdoor:EC2/C&CActivity.B	TTPs/Command and Control/Backdoor:EC2-C&CActivity.B
Backdoor:EC2/C&CActivity.B!DNS	TTPs/Command and Control/Backdoor:EC2-C&CActivity.B!DNS
Backdoor:EC2/DenialOfService.Dns	TTPs/Command and Control/Backdoor:EC2-DenialOfService.Dns
Backdoor:EC2/DenialOfService.Tcp	TTPs/Command and Control/Backdoor:EC2-DenialOfService.Tcp

GuardDuty finding type	ASFF finding type
Backdoor:EC2/DenialOfService.Udp	TTPs/Command and Control/Backdoor:EC2-DenialOfService.Udp
Backdoor:EC2/DenialOfService.UdpOnTcpPorts	TTPs/Command and Control/Backdoor:EC2-DenialOfService.UdpOnTcpPorts
Backdoor:EC2/DenialOfService.UnusualProtocol	TTPs/Command and Control/Backdoor:EC2-DenialOfService.UnusualProtocol
Backdoor:EC2/Spambot	TTPs/Command and Control/Backdoor:EC2-Spambot
Behavior:EC2/NetworkPortUnusual	Unusual Behaviors/VM/Behavior:EC2-NetworkPortUnusual
Behavior:EC2/TrafficVolumeUnusual	Unusual Behaviors/VM/Behavior:EC2-TrafficVolumeUnusual
Backdoor:Lambda/C&CActivity.B	TTPs/Command and Control/Backdoor:Lambda-C&CActivity.B
Backdoor:Runtime/C&CActivity.B	TTPs/Command and Control/Backdoor:Runtime-C&CActivity.B
Backdoor:Runtime/C&CActivity.B!DNS	TTPs/Command and Control/Backdoor:Runtime-C&CActivity.B!DNS
CredentialAccess:IAMUser/AnomalousBehavior	TTPs/Credential Access/IAMUser-AnomalousBehavior
CredentialAccess:Kubernetes/AnomalousBehavior.SecretsAccessed	TTPs/AnomalousBehavior/CredentialAccess:Kubernetes-SecretsAccessed
CredentialAccess:Kubernetes/MaliciousIPCaller	TTPs/CredentialAccess/CredentialAccess:Kubernetes-MaliciousIPCaller
CredentialAccess:Kubernetes/MaliciousIPCaller.Custom	TTPs/CredentialAccess/CredentialAccess:Kubernetes-MaliciousIPCaller.Custom

GuardDuty finding type	ASFF finding type
CredentialAccess:Kubernetes/SuccessfulAnonymousAccess	TTPs/CredentialAccess/CredentialAccess:Kubernetes-SuccessfulAnonymousAccess
CredentialAccess:Kubernetes/TorIPCaller	TTPs/CredentialAccess/CredentialAccess:Kubernetes-TorIPCaller
CredentialAccess:RDS/AnomalousBehavior.FailedLogin	TTPs/Credential Access/CredentialAccess:RDS-AnomalousBehavior.FailedLogin
CredentialAccess:RDS/AnomalousBehavior.SuccessfulBruteForce	TTPs/Credential Access/RDS-AnomalousBehavior.SuccessfulBruteForce
CredentialAccess:RDS/AnomalousBehavior.SuccessfulLogin	TTPs/Credential Access/RDS-AnomalousBehavior.SuccessfulLogin
CredentialAccess:RDS/MaliciousIPCaller.FailedLogin	TTPs/Credential Access/RDS-MaliciousIPCaller.FailedLogin
CredentialAccess:RDS/MaliciousIPCaller.SuccessfulLogin	TTPs/Credential Access/RDS-MaliciousIPCaller.SuccessfulLogin
CredentialAccess:RDS/TorIPCaller.FailedLogin	TTPs/Credential Access/RDS-TorIPCaller.FailedLogin
CredentialAccess:RDS/TorIPCaller.SuccessfulLogin	TTPs/Credential Access/RDS-TorIPCaller.SuccessfulLogin
CryptoCurrency:EC2/BitcoinTool.B	TTPs/Command and Control/CryptoCurrency:EC2-BitcoinTool.B
CryptoCurrency:EC2/BitcoinTool.B!DNS	TTPs/Command and Control/CryptoCurrency:EC2-BitcoinTool.B!DNS
CryptoCurrency:Lambda/BitcoinTool.B	TTPs/Command and Control/CryptoCurrency:Lambda-BitcoinTool.B Effects/Resource Consumption/CryptoCurrency:Lambda-BitcoinTool.B

GuardDuty finding type	ASFF finding type
CryptoCurrency:Runtime/BitcoinTool.B	TTPs/Command and Control/CryptoCurrency:Runtime-BitcoinTool.B
CryptoCurrency:Runtime/BitcoinTool.B!DNS	TTPs/Command and Control/CryptoCurrency:Runtime-BitcoinTool.B!DNS
DefenseEvasion:EC2/UnusualDNSResolver	TTPs/DefenseEvasion/EC2:Unusual-DNS-Resolver
DefenseEvasion:EC2/UnusualDoHActivity	TTPs/DefenseEvasion/EC2:Unusual-DoH-Activity
DefenseEvasion:EC2/UnusualDoTActivity	TTPs/DefenseEvasion/EC2:Unusual-DoT-Activity
DefenseEvasion:IAMUser/AnomalousBehavior	TTPs/Defense Evasion/IAMUser-AnomalousBehavior
DefenseEvasion:Kubernetes/MaliciousIPCaller	TTPs/DefenseEvasion/DefenseEvasion:Kubernetes-MaliciousIPCaller
DefenseEvasion:Kubernetes/MaliciousIPCaller.Custom	TTPs/DefenseEvasion/DefenseEvasion:Kubernetes-MaliciousIPCaller.Custom
DefenseEvasion:Kubernetes/SuccessfulAnonymousAccess	TTPs/DefenseEvasion/DefenseEvasion:Kubernetes-SuccessfulAnonymousAccess
DefenseEvasion:Kubernetes/TorIPCaller	TTPs/DefenseEvasion/DefenseEvasion:Kubernetes-TorIPCaller
DefenseEvasion:Runtime/FilelessExecution	TTPs/Defense Evasion/DefenseEvasion:Runtime-FilelessExecution
DefenseEvasion:Runtime/ProcessInjection.Proc	TTPs/Defense Evasion/DefenseEvasion:Runtime-ProcessInjection.Proc
DefenseEvasion:Runtime/ProcessInjection.Ptrace	TTPs/Defense Evasion/DefenseEvasion:Runtime-ProcessInjection.Ptrace

GuardDuty finding type	ASFF finding type
DefenseEvasion:Runtime/ProcessInjection.VirtualMemoryWrite	TTPs/Defense Evasion/DefenseEvasion:Runtime-ProcessInjection.VirtualMemoryWrite
DefenseEvasion:Runtime/PtraceAntiDebugging	TTPs/DefenseEvasion/DefenseEvasion:Runtime-PtraceAntiDebugging
DefenseEvasion:Runtime/SuspiciousCommand	TTPs/DefenseEvasion/DefenseEvasion:Runtime-SuspiciousCommand
Discovery:IAMUser/AnomalousBehavior	TTPs/Discovery/IAMUser-AnomalousBehavior
Discovery:Kubernetes/AnomalousBehavior.PermissionChecked	TTPs/AnomalousBehavior/Discovery:Kubernetes-PermissionChecked
Discovery:Kubernetes/MaliciousIPCaller	TTPs/Discovery/Discovery:Kubernetes-MaliciousIPCaller
Discovery:Kubernetes/MaliciousIPCaller.Custom	TTPs/Discovery/Discovery:Kubernetes-MaliciousIPCaller.Custom
Discovery:Kubernetes/SuccessfulAnonymousAccess	TTPs/Discovery/Discovery:Kubernetes-SuccessfulAnonymousAccess
Discovery:Kubernetes/TorIPCaller	TTPs/Discovery/Discovery:Kubernetes-TorIPCaller
Discovery:RDS/MaliciousIPCaller	TTPs/Discovery/RDS-MaliciousIPCaller
Discovery:RDS/TorIPCaller	TTPs/Discovery/RDS-TorIPCaller
Discovery:Runtime/SuspiciousCommand	TTPs/Discovery/Discovery:Runtime-SuspiciousCommand
Discovery:S3/AnomalousBehavior	TTPs/Discovery:S3-AnomalousBehavior
Discovery:S3/BucketEnumeration.Unusual	TTPs/Discovery:S3-BucketEnumeration.Unusual

GuardDuty finding type	ASFF finding type
Discovery:S3/MaliciousIPCaller.Custom	TTPs/Discovery:S3-MaliciousIPCaller.Custom
Discovery:S3/TorIPCaller	TTPs/Discovery:S3-TorIPCaller
Discovery:S3/MaliciousIPCaller	TTPs/Discovery:S3-MaliciousIPCaller
Exfiltration:IAMUser/AnomalousBehavior	TTPs/Exfiltration/IAMUser-AnomalousBehavior
Execution:Kubernetes/ExecInKubeSystemPod	TTPs/Execution/Execution:Kubernetes-ExecInKubeSystemPod
Execution:Kubernetes/AnomalousBehavior.ExecInPod	TTPs/AnomalousBehavior/Execution:Kubernetes-ExecInPod
Execution:Kubernetes/AnomalousBehavior.WorkloadDeployed	TTPs/AnomalousBehavior/Execution:Kubernetes-WorkloadDeployed
Impact:EC2/MaliciousDomainRequest.Custom	TTPs/Impact/Impact:EC2-MaliciousDomainRequest.Custom
Impact:Kubernetes/MaliciousIPCaller	TTPs/Impact/Impact:Kubernetes-MaliciousIPCaller
Impact:Kubernetes/MaliciousIPCaller.Custom	TTPs/Impact/Impact:Kubernetes-MaliciousIPCaller.Custom
Impact:Kubernetes/SuccessfulAnonymousAccess	TTPs/Impact/Impact:Kubernetes-SuccessfulAnonymousAccess
Impact:Kubernetes/TorIPCaller	TTPs/Impact/Impact:Kubernetes-TorIPCaller
Persistence:Kubernetes/ContainerWithSensitiveMount	TTPs/Persistence/Persistence:Kubernetes-ContainerWithSensitiveMount
Persistence:Kubernetes/AnomalousBehavior.WorkloadDeployed!ContainerWithSensitiveMount	TTPs/AnomalousBehavior/Persistence:Kubernetes-WorkloadDeployed!ContainerWithSensitiveMount

GuardDuty finding type	ASFF finding type
PrivilegeEscalation:Kubernetes/AnomalousBehavior.WorkloadDeployed!PrivilegedContainer	TTPs/AnomalousBehavior/PrivilegeEscalation:Kubernetes-WorkloadDeployed!PrivilegedContainer
Persistence:Kubernetes/MaliciousIPCaller	TTPs/Persistence/Persistence:Kubernetes-MaliciousIPCaller
Persistence:Kubernetes/MaliciousIPCaller.Custom	TTPs/Persistence/Persistence:Kubernetes-MaliciousIPCaller.Custom
Persistence:Kubernetes/SuccessfulAnonymousAccess	TTPs/Persistence/Persistence:Kubernetes-SuccessfulAnonymousAccess
Persistence:Kubernetes/TorIPCaller	TTPs/Persistence/Persistence:Kubernetes-TorIPCaller
Execution:EC2/MaliciousFile	TTPs/Execution/Execution:EC2-MaliciousFile
Execution:ECS/MaliciousFile	TTPs/Execution/Execution:ECS-MaliciousFile
Execution:Kubernetes/MaliciousFile	TTPs/Execution/Execution:Kubernetes-MaliciousFile
Execution:Container/MaliciousFile	TTPs/Execution/Execution:Container-MaliciousFile
Execution:EC2/SuspiciousFile	TTPs/Execution/Execution:EC2-SuspiciousFile
Execution:ECS/SuspiciousFile	TTPs/Execution/Execution:ECS-SuspiciousFile
Execution:Kubernetes/SuspiciousFile	TTPs/Execution/Execution:Kubernetes-SuspiciousFile
Execution:Container/SuspiciousFile	TTPs/Execution/Execution:Container-SuspiciousFile
Execution:Runtime/MaliciousFileExecuted	TTPs/Execution/Execution:Runtime-MaliciousFileExecuted

GuardDuty finding type	ASFF finding type
Execution:Runtime/NewBinaryExecuted	TTPs/Execution/Execution:Runtime-NewBinaryExecuted
Execution:Runtime/NewLibraryLoaded	TTPs/Execution/Execution:Runtime-NewLibraryLoaded
Execution:Runtime/ReverseShell	TTPs/Execution/Execution:Runtime-ReverseShell
Execution:Runtime/SuspiciousCommand	TTPs/Execution/Execution:Runtime-SuspiciousCommand
Execution:Runtime/SuspiciousShellCreated	TTPs/Execution/Execution:Runtime-SuspiciousShellCreated
Execution:Runtime/SuspiciousTool	TTPs/Execution/Execution:Runtime-SuspiciousTool
Exfiltration:S3/AnomalousBehavior	TTPs/Exfiltration:S3-AnomalousBehavior
Exfiltration:S3/ObjectRead.Unusual	TTPs/Exfiltration:S3-ObjectRead.Unusual
Exfiltration:S3/MaliciousIPCaller	TTPs/Exfiltration:S3-MaliciousIPCaller
Impact:EC2/AbusedDomainRequest.Reputation	TTPs/Impact:EC2-AbusedDomainRequest.Reputation
Impact:EC2/BitcoinDomainRequest.Reputation	TTPs/Impact:EC2-BitcoinDomainRequest.Reputation
Impact:EC2/MaliciousDomainRequest.Reputation	TTPs/Impact:EC2-MaliciousDomainRequest.Reputation
Impact:EC2/PortSweep	TTPs/Impact/Impact:EC2-PortSweep
Impact:EC2/SuspiciousDomainRequest.Reputation	TTPs/Impact:EC2-SuspiciousDomainRequest.Reputation

GuardDuty finding type	ASFF finding type
Impact:EC2/WinRMBruteForce	TTPs/Impact/Impact:EC2-WinRMBruteForce
Impact:IAMUser/AnomalousBehavior	TTPs/Impact/IAMUser-AnomalousBehavior
Impact:Runtime/AbusedDomainRequest.Reputation	TTPs/Impact/Impact:Runtime-AbusedDomainRequest.Reputation
Impact:Runtime/BitcoinDomainRequest.Reputation	TTPs/Impact/Impact:Runtime-BitcoinDomainRequest.Reputation
Impact:Runtime/CryptoMinerExecuted	TTPs/Impact/Impact:Runtime-CryptoMinerExecuted
Impact:Runtime/MaliciousDomainRequest.Reputation	TTPs/Impact/Impact:Runtime-MaliciousDomainRequest.Reputation
Impact:Runtime/SuspiciousDomainRequest.Reputation	TTPs/Impact/Impact:Runtime-SuspiciousDomainRequest.Reputatio
Impact:S3/AnomalousBehavior.Delete	TTPs/Impact:S3-AnomalousBehavior.Delete
Impact:S3/AnomalousBehavior.Permission	TTPs/Impact:S3-AnomalousBehavior.Permission
Impact:S3/AnomalousBehavior.Write	TTPs/Impact:S3-AnomalousBehavior.Write
Impact:S3/ObjectDelete.Unusual	TTPs/Impact:S3-ObjectDelete.Unusual
Impact:S3/PermissionsModification.Unusual	TTPs/Impact:S3-PermissionsModification.Unusual
Impact:S3/MaliciousIPCaller	TTPs/Impact:S3-MaliciousIPCaller
InitialAccess:IAMUser/AnomalousBehavior	TTPs/Initial Access/IAMUser-AnomalousBehavior
Object:S3/MaliciousFile	TTPs/Object/Object:S3-MaliciousFile
PenTest:IAMUser/KaliLinux	TTPs/PenTest:IAMUser/KaliLinux

GuardDuty finding type	ASFF finding type
PenTest:IAMUser/ParrotLinux	TTPs/PenTest:IAMUser/ParrotLinux
PenTest:IAMUser/PentooLinux	TTPs/PenTest:IAMUser/PentooLinux
PenTest:S3/KaliLinux	TTPs/PenTest:S3-KaliLinux
PenTest:S3/ParrotLinux	TTPs/PenTest:S3-ParrotLinux
PenTest:S3/PentooLinux	TTPs/PenTest:S3-PentooLinux
Persistence:IAMUser/AnomalousBehavior	TTPs/Persistence/IAMUser-AnomalousBehavior
Persistence:IAMUser/NetworkPermissions	TTPs/Persistence/Persistence:IAMUser-NetworkPermissions
Persistence:IAMUser/ResourcePermissions	TTPs/Persistence/Persistence:IAMUser-ResourcePermissions
Persistence:IAMUser/UserPermissions	TTPs/Persistence/Persistence:IAMUser-UserPermissions
Persistence:Runtime/SuspiciousCommand	TTPs/Persistence/Persistence:Runtime-SuspiciousCommand
Policy:IAMUser/RootCredentialUsage	TTPs/Policy:IAMUser-RootCredentialUsage
Policy:IAMUser/ShortTermRootCredentialUsage	TTPs/Policy:IAMUser-ShortTermRootCredentialUsage
Policy:Kubernetes/AdminAccessToDefaultServiceAccount	Software and Configuration Checks/AWS Security Best Practices/Policy:Kubernetes-AdminAccessToDefaultServiceAccount
Policy:Kubernetes/AnonymousAccessGranted	Software and Configuration Checks/AWS Security Best Practices/Policy:Kubernetes-AnonymousAccessGranted

GuardDuty finding type	ASFF finding type
Policy:Kubernetes/ExposedDashboard	Software and Configuration Checks/AWS Security Best Practices/Policy:Kubernetes-ExposedDashboard
Policy:Kubernetes/KubeflowDashboardExposed	Software and Configuration Checks/AWS Security Best Practices/Policy:Kubernetes-KubeflowDashboardExposed
Policy:S3/AccountBlockPublicAccessDisabled	TTPs/Policy:S3-AccountBlockPublicAccessDisabled
Policy:S3/BucketAnonymousAccessGranted	TTPs/Policy:S3-BucketAnonymousAccessGranted
Policy:S3/BucketBlockPublicAccessDisabled	Effects/Data Exposure/Policy:S3-BucketBlockPublicAccessDisabled
Policy:S3/BucketPublicAccessGranted	TTPs/Policy:S3-BucketPublicAccessGranted
PrivilegeEscalation:IAMUser/AnomalousBehavior	TTPs/Privilege Escalation/IAMUser-AnomalousBehavior
PrivilegeEscalation:IAMUser/AdministrativePermissions	TTPs/Privilege Escalation/PrivilegeEscalation:IAMUser-AdministrativePermissions
PrivilegeEscalation:Kubernetes/AnomalousBehavior.RoleBindingCreated	TTPs/AnomalousBehavior/PrivilegeEscalation:Kubernetes-RoleBindingCreated
PrivilegeEscalation:Kubernetes/AnomalousBehavior.RoleCreated	TTPs/AnomalousBehavior/PrivilegeEscalation:Kubernetes-RoleCreated
PrivilegeEscalation:Kubernetes/PrivilegedContainer	TTPs/PrivilegeEscalation/PrivilegeEscalation:Kubernetes-PrivilegedContainer
PrivilegeEscalation:Runtime/ContainerMountsHostDirectory	TTPs/Privilege Escalation/PrivilegeEscalation:Runtime-ContainerMountsHostDirectory

GuardDuty finding type	ASFF finding type
PrivilegeEscalation:Runtime/CGroupsReleaseAgentModified	TTPs/Privilege Escalation/PrivilegeEscalation:Runtime-CGroupsReleaseAgentModified
PrivilegeEscalation:Runtime/DockerSocketAccessed	TTPs/Privilege Escalation/PrivilegeEscalation:Runtime-DockerSocketAccessed
PrivilegeEscalation:Runtime/ElevationToRoot	TTPs/Privilege Escalation/PrivilegeEscalation:Runtime-ElevationToRoot
PrivilegeEscalation:Runtime/RuncContainerEscape	TTPs/Privilege Escalation/PrivilegeEscalation:Runtime-RuncContainerEscape
PrivilegeEscalation:Runtime/SuspiciousCommand	Software and Configuration Checks/PrivilegeEscalation:Runtime-SuspiciousCommand
PrivilegeEscalation:Runtime/UserfaultfdUsage	TTPs/Privilege Escalation/PrivilegeEscalation:Runtime-UserfaultfdUsage
Recon:EC2/PortProbeEMRUnprotectedPort	TTPs/Discovery/Recon:EC2-PortProbeEMRUnprotectedPort
Recon:EC2/PortProbeUnprotectedPort	TTPs/Discovery/Recon:EC2-PortProbeUnprotectedPort
Recon:EC2/Portscan	TTPs/Discovery/Recon:EC2-Portscan
Recon:IAMUser/MaliciousIPCaller	TTPs/Discovery/Recon:IAMUser-MaliciousIPCaller
Recon:IAMUser/MaliciousIPCaller.Custom	TTPs/Discovery/Recon:IAMUser-MaliciousIPCaller.Custom
Recon:IAMUser/NetworkPermissions	TTPs/Discovery/Recon:IAMUser-NetworkPermissions
Recon:IAMUser/ResourcePermissions	TTPs/Discovery/Recon:IAMUser-ResourcePermissions

GuardDuty finding type	ASFF finding type
Recon:IAMUser/TorIPCaller	TTPs/Discovery/Recon:IAMUser-TorIPCaller
Recon:IAMUser/UserPermissions	TTPs/Discovery/Recon:IAMUser-UserPer missions
ResourceConsumption:IAMUser/ComputeR esources	Unusual Behaviors/User/ResourceCons umption:IAMUser-ComputeResources
Stealth:IAMUser/CloudTrailLoggingDisabled	TTPs/Defense Evasion/Stealth:IAMUser-Clo udTrailLoggingDisabled
Stealth:IAMUser/LoggingConfiguration Modified	TTPs/Defense Evasion/Stealth:IAMUser-Log gingConfigurationModified
Stealth:IAMUser/PasswordPolicyChange	TTPs/Defense Evasion/Stealth:IAMUser-Pas swordPolicyChange
Stealth:S3/ServerAccessLoggingDisabled	TTPs/Defense Evasion/Stealth:S3-ServerAc cessLoggingDisabled
Trojan:EC2/BlackholeTraffic	TTPs/Command and Control/Trojan:EC2- BlackholeTraffic
Trojan:EC2/BlackholeTraffic!DNS	TTPs/Command and Control/Trojan:EC2- BlackholeTraffic!DNS
Trojan:EC2/DGADomainRequest.B	TTPs/Command and Control/Trojan:EC2- DGADomainRequest.B
Trojan:EC2/DGADomainRequest.C!DNS	TTPs/Command and Control/Trojan:EC2- DGADomainRequest.C!DNS
Trojan:EC2/DNSDataExfiltration	TTPs/Command and Control/Trojan:EC2- DNSDataExfiltration
Trojan:EC2/DriveBySourceTraffic!DNS	TTPs/Initial Access/Trojan:EC2-DriveBySo urceTraffic!DNS

GuardDuty finding type	ASFF finding type
Trojan:EC2/DropPoint	Effects/Data Exfiltration/Trojan:EC2-DropPoint
Trojan:EC2/DropPoint!DNS	Effects/Data Exfiltration/Trojan:EC2-DropPoint!DNS
Trojan:EC2/PhishingDomainRequest!DNS	TTPs/Command and Control/Trojan:EC2-PhishingDomainRequest!DNS
Trojan:Lambda/BlackholeTraffic	TTPs/Command and Control/Trojan:Lambda-BlackholeTraffic
Trojan:Lambda/DropPoint	Effects/Data Exfiltration/Trojan:Lambda-DropPoint
Trojan:Runtime/BlackholeTraffic	TTPs/Command and Control/Trojan:Runtime-BlackholeTraffic
Trojan:Runtime/BlackholeTraffic!DNS	TTPs/Command and Control/Trojan:Runtime-BlackholeTraffic!DNS
Trojan:Runtime/DGADomainRequest.C!DNS	TTPs/Command and Control/Trojan:Runtime-DGADomainRequest.C!DNS
Trojan:Runtime/DriveBySourceTraffic!DNS	TTPs/Initial Access/Trojan:Runtime-DriveBySourceTraffic!DNS
Trojan:Runtime/DropPoint	Effects/Data Exfiltration/Trojan:Runtime-DropPoint
Trojan:Runtime/DropPoint!DNS	Effects/Data Exfiltration/Trojan:Runtime-DropPoint!DNS
Trojan:Runtime/PhishingDomainRequest!DNS	TTPs/Command and Control/Trojan:Runtime-PhishingDomainRequest!DNS
UnauthorizedAccess:EC2/MaliciousIPCaller.Custom	TTPs/Command and Control/UnauthorizedAccess:EC2-MaliciousIPCaller.Custom

GuardDuty finding type	ASFF finding type
UnauthorizedAccess:EC2/MetadataDNSRebind	TTPs/UnauthorizedAccess:EC2-Metadata DNSRebind
UnauthorizedAccess:EC2/RDPBruteForce	TTPs/Initial Access/UnauthorizedAccess:EC2-RDPBruteForce
UnauthorizedAccess:EC2/SSHBruteForce	TTPs/Initial Access/UnauthorizedAccess:EC2-SSHBruteForce
UnauthorizedAccess:EC2/TorClient	Effects/Resource Consumption/Unauth orizedAccess:EC2-TorClient
UnauthorizedAccess:EC2/TorRelay	Effects/Resource Consumption/Unauth orizedAccess:EC2-TorRelay
UnauthorizedAccess:IAMUser/ConsoleLogin	Unusual Behaviors/User/Unauthorized Access:IAMUser-ConsoleLogin
UnauthorizedAccess:IAMUser/ConsoleLo ginSuccess.B	TTPs/UnauthorizedAccess:IAMUser-Cons oleLoginSuccess.B
UnauthorizedAccess:IAMUser/InstanceC redentialExfiltration.InsideAWS	Effects/Data Exfiltration/UnauthorizedAc cess:IAMUser-InstanceCredentialExfiltration.I nsideAWS
UnauthorizedAccess:IAMUser/InstanceC redentialExfiltration.OutsideAWS	Effects/Data Exfiltration/UnauthorizedAc cess:IAMUser-InstanceCredentialExfiltration.O utsideAWS
UnauthorizedAccess:IAMUser/Malicious IPCaller	TTPs/UnauthorizedAccess:IAMUser-Mali ciousIPCaller
UnauthorizedAccess:IAMUser/Malicious IPCaller.Custom	TTPs/UnauthorizedAccess:IAMUser-Mali ciousIPCaller.Custom
UnauthorizedAccess:IAMUser/TorIPCaller	TTPs/Command and Control/Unauthoriz edAccess:IAMUser-TorIPCaller

GuardDuty finding type	ASFF finding type
UnauthorizedAccess:Lambda/MaliciousIPCaller.Custom	TTPs/Command and Control/UnauthorizedAccess:Lambda-MaliciousIPCaller.Custom
UnauthorizedAccess:Lambda/TorClient	Effects/Resource Consumption/UnauthorizedAccess:Lambda-TorClient
UnauthorizedAccess:Lambda/TorRelay	Effects/Resource Consumption/UnauthorizedAccess:Lambda-TorRelay
UnauthorizedAccess:Runtime/MetadataDNSRebind	TTPs/UnauthorizedAccess:Runtime-MetadataDNSRebind
UnauthorizedAccess:Runtime/TorRelay	Effects/Resource Consumption/UnauthorizedAccess:Runtime-TorRelay
UnauthorizedAccess:Runtime/TorClient	Effects/Resource Consumption/UnauthorizedAccess:Runtime-TorClient
UnauthorizedAccess:S3/MaliciousIPCaller.Custom	TTPs/UnauthorizedAccess:S3-MaliciousIPCaller.Custom
UnauthorizedAccess:S3/TorIPCaller	TTPs/UnauthorizedAccess:S3-TorIPCaller

Typical finding from GuardDuty

GuardDuty sends findings to Security Hub using the AWS Security Finding Format (ASFF).

Here is an example of a typical finding from GuardDuty.


```
{
  "SchemaVersion": "2018-10-08",
  "Id": "arn:aws:guardduty:us-east-1:193043430472:detector/
d4b040365221be2b54a6264dc9a4bc64/finding/46ba0ac2845071e23ccdeb2ae03bfdea",
  "ProductArn": "arn:aws:securityhub:us-east-1:product/aws/guardduty",
  "GeneratorId": "arn:aws:guardduty:us-east-1:193043430472:detector/
d4b040365221be2b54a6264dc9a4bc64",
  "AwsAccountId": "193043430472",
```

```
  "Types": [
    "TTPs/Initial Access/UnauthorizedAccess:EC2-SSHBruteForce"
  ],
  "FirstObservedAt": "2020-08-22T09:15:57Z",
  "LastObservedAt": "2020-09-30T11:56:49Z",
  "CreatedAt": "2020-08-22T09:34:34.146Z",
  "UpdatedAt": "2020-09-30T12:14:00.206Z",
  "Severity": {
    "Product": 2,
    "Label": "MEDIUM",
    "Normalized": 40
  },
  "Title": "199.241.229.197 is performing SSH brute force attacks against
 i-0c10c2c7863d1a356.",
  "Description": "199.241.229.197 is performing SSH brute force attacks against
 i-0c10c2c7863d1a356. Brute force attacks are used to gain unauthorized access to your
 instance by guessing the SSH password.",
  "SourceUrl": "https://us-east-1.console.aws.amazon.com/guardduty/home?region=us-
east-1#/findings?macros=current&fId=46ba0ac2845071e23ccdeb2ae03bfdea",
  "ProductFields": {
    "aws/guardduty/service/action/networkConnectionAction/remotePortDetails/portName":
 "Unknown",
    "aws/guardduty/service/archived": "false",
    "aws/guardduty/service/action/networkConnectionAction/remoteIpDetails/organization/
asnOrg": "CENTURYLINK-US-LEGACY-QWEST",
    "aws/guardduty/service/action/networkConnectionAction/remoteIpDetails/geoLocation/
lat": "42.5122",
    "aws/guardduty/service/action/networkConnectionAction/remoteIpDetails/ipAddressV4":
 "199.241.229.197",
    "aws/guardduty/service/action/networkConnectionAction/remoteIpDetails/geoLocation/
lon": "-90.7384",
    "aws/guardduty/service/action/networkConnectionAction/blocked": "false",
    "aws/guardduty/service/action/networkConnectionAction/remotePortDetails/port":
 "46717",
    "aws/guardduty/service/action/networkConnectionAction/remoteIpDetails/country/
countryName": "United States",
    "aws/guardduty/service/serviceName": "guardduty",
    "aws/guardduty/service/evidence": "",
    "aws/guardduty/service/action/networkConnectionAction/localIpDetails/ipAddressV4":
 "172.31.43.6",
    "aws/guardduty/service/detectorId": "d4b040365221be2b54a6264dc9a4bc64",
    "aws/guardduty/service/action/networkConnectionAction/remoteIpDetails/organization/
org": "CenturyLink",
```

```
    "aws/guardduty/service/action/networkConnectionAction/connectionDirection":
 "INBOUND",
    "aws/guardduty/service/eventFirstSeen": "2020-08-22T09:15:57Z",
    "aws/guardduty/service/eventLastSeen": "2020-09-30T11:56:49Z",
    "aws/guardduty/service/action/networkConnectionAction/localPortDetails/portName":
 "SSH",
    "aws/guardduty/service/action/actionType": "NETWORK_CONNECTION",
    "aws/guardduty/service/action/networkConnectionAction/remoteIpDetails/city/
cityName": "Dubuque",
    "aws/guardduty/service/additionalInfo": "",
    "aws/guardduty/service/resourceRole": "TARGET",
    "aws/guardduty/service/action/networkConnectionAction/localPortDetails/port": "22",
    "aws/guardduty/service/action/networkConnectionAction/protocol": "TCP",
    "aws/guardduty/service/count": "74",
    "aws/guardduty/service/action/networkConnectionAction/remoteIpDetails/organization/
asn": "209",
    "aws/guardduty/service/action/networkConnectionAction/remoteIpDetails/organization/
isp": "CenturyLink",
    "aws/securityhub/FindingId": "arn:aws:securityhub:us-east-1::product/aws/guardduty/
arn:aws:guardduty:us-east-1:193043430472:detector/d4b040365221be2b54a6264dc9a4bc64/
finding/46ba0ac2845071e23ccdeb2ae03bfdea",
    "aws/securityhub/ProductName": "GuardDuty",
    "aws/securityhub/CompanyName": "Amazon"
  },
  "Resources": [
    {
      "Type": "AwsEc2Instance",
      "Id": "arn:aws:ec2:us-east-1:193043430472:instance/i-0c10c2c7863d1a356",
      "Partition": "aws",
      "Region": "us-east-1",
      "Tags": {
        "Name": "kubectl"
      },
      "Details": {
        "AwsEc2Instance": {
          "Type": "t2.micro",
          "ImageId": "ami-02354e95b39ca8dec",
          "IpV4Addresses": [
            "18.234.130.16",
            "172.31.43.6"
          ],
          "VpcId": "vpc-a0c2d7c7",
          "SubnetId": "subnet-4975b475",
          "LaunchedAt": "2020-08-03T23:21:57Z"
```

```
        }
      }
    }
  ],
  "WorkflowState": "NEW",
  "Workflow": {
    "Status": "NEW"
  },
  "RecordState": "ACTIVE"
}
```

Enabling and configuring the integration

To use the integration with AWS Security Hub, you must enable Security Hub. For information on how to enable Security Hub, see Setting up Security Hub in the *AWS Security Hub User Guide*.

When you enable both GuardDuty and Security Hub, the integration is enabled automatically. GuardDuty immediately begins to send findings to Security Hub.

Using GuardDuty controls in Security Hub

AWS Security Hub uses security controls to evaluate your AWS resources, and check your compliance against security industry standards and best practices. You can use the controls related to GuardDuty resources and selected protection plans. For more information, see Amazon GuardDuty controls in the *AWS Security Hub User Guide*.

For a list of all the controls across AWS services and resources, see Security Hub controls reference in the *AWS Security Hub User Guide*.

Stopping the publication of findings to Security Hub

To stop sending findings to Security Hub, you can use either the Security Hub console or the API.

See Disabling and enabling the flow of findings from an integration (console) or Disabling the flow of findings from an integration (Security Hub API, AWS CLI) in the *AWS Security Hub User Guide*.

Integrating with Amazon Detective

Amazon Detective helps you quickly analyze and investigate security events across one or more AWS accounts by generating data visualizations that represent the ways your resources behave and interact over time. Detective creates visualizations of GuardDuty findings.

Detective ingests finding details for all finding types, and provides access to the entity profiles to investigate different entities that are involved with the finding. An entity can be an AWS account, an AWS resource within an account, or an external IP Address that has interacted with your resources. The GuardDuty console supports pivoting to Amazon Detective from the following entities, depending on finding type: AWS account, IAM role, user, or role session, user agent, federated user, Amazon EC2 instance, or IP address.

Contents

- Enabling the integration
- Pivoting to Amazon Detective from a GuardDuty finding
- Using the integration with a GuardDuty multi-account environment

Enabling the integration

To use Amazon Detective with GuardDuty you must first enable Amazon Detective. For information on how to enable Detective, see Geting started with Amazon Detective in the *Amazon Detective User Guide*.

When you enable both GuardDuty and Detective, the integration is enabled automatically. Once enabled, Detective will immediately ingest your GuardDuty findings data.

> **Note**
>
> GuardDuty sends findings to Detective based on the GuardDuty findings export frequency. By default, the export frequency for updates to existing findings is 6 hours. To ensure Detective receives the most recent updates to your findings it is recommended that you change the export frequency to 15 minutes in each region in which you use Detective with GuardDuty. For more information see Step 5 – Setting frequency to export updated active findings.

Pivoting to Amazon Detective from a GuardDuty finding

1. Log into the https://console.aws.amazon.com/guardduty/ console.
2. Choose a single finding from your findings table.
3. Choose **Investigate with Detective** from the finding details pane.

4. Choose an aspect of the finding to investigate with Amazon Detective. This opens the Detective console for that finding or entity.

If the pivot does not behave as expected, see Troubleshooting the pivot in the *Amazon Detective User Guide*.

> **Note**
>
> If you archive a GuardDuty finding in the Detective console, that finding gets archived in the GuardDuty console as well.

Using the integration with a GuardDuty multi-account environment

If you are managing a multi-account environment in GuardDuty, you must add your member accounts to Amazon Detective to view Detective data visualizations for findings and entities in those accounts.

It is recommended that you use the same GuardDuty Administrator account as the administrator account for Detective. For more information on adding member accounts in Detective, see Managing accounts in the *Amazon Detective User Guide*.

> **Note**
>
> Detective is a regional service, meaning you must enable Detective and add your member accounts in each region in which you want to use the integration.

Suspending or disabling GuardDuty

You can use the GuardDuty console to suspend or disable the GuardDuty service. You don't get charged for using GuardDuty when the service is suspended.

- All member accounts must be disassociated or deleted before you can suspend or disable GuardDuty.
- If you suspend GuardDuty, it no longer monitors the security of your AWS environment or generates new findings. Your existing findings remain intact and are not affected by the GuardDuty suspension. You can choose to re-enable GuardDuty later.
- When you disable GuardDuty in an account, it will be disabled only for the currently selected AWS Region. If you want to completely disable GuardDuty, you must disable it in each Region where it is enabled.
- If you disable GuardDuty, your existing findings and the GuardDuty configuration are lost and can't be recovered. If you want to save your existing findings, you must export them before you confirm to disable GuardDuty. For information on how to export findings, see Exporting generated findings to Amazon S3.
- If you have enabled Malware Protection for S3 for one or more protected buckets in your account, then suspending or disabling GuardDuty doesn't impact the status of a protected bucket under Malware Protection for S3. Even after suspending or disabling GuardDuty, your account will continue incurring the usage costs associated with the Malware Protection for S3 feature. For information about disabling Malware Protection for S3, see Disabling Malware Protection for S3 for a protected bucket.

To suspend or disable GuardDuty

1. Open the GuardDuty console at https://console.aws.amazon.com/guardduty/.
2. In the navigation pane, choose **Settings**.
3. In the **Suspend GuardDuty** section, choose **Suspend GuardDuty** or **Disable GuardDuty**, then **Confirm** your action.

To re-enable GuardDuty after suspending

1. Open the GuardDuty console at https://console.aws.amazon.com/guardduty/.
2. In the navigation pane, choose **Settings**.

3. Choose **Re-enable GuardDuty**.

Subscribing to Amazon SNS GuardDuty announcements

This section provides information about subscribing to Amazon SNS (Simple Notification Service) for GuardDuty announcements to receive notifications about newly released finding types, updates to the existing finding types, and other functionality changes. Notifications are available in all formats that Amazon SNS supports.

The GuardDuty SNS sends announcement about updates to the GuardDuty service across AWS to any subscribed account. To receive notifications about findings within your account, see Processing GuardDuty findings with Amazon EventBridge.

> **Note**
>
> Your IAM user must have `sns::subscribe` permissions to subscribe to an SNS.

You can subscribe an Amazon SQS queue to this notification topic, but you must use a topic ARN that is in the same Region. For more information, see Tutorial: Subscribing an Amazon SQS queue to an Amazon SNS topic in the *Amazon Simple Queue Service developer guide*.

You can also use an AWS Lambda function to trigger events when notifications are received. For more information, see Invoking Lambda functions using Amazon SNS notifications in the *Amazon Simple Queue Service developer guide*.

The Amazon SNS topic ARNs for each Region are shown below.

AWS Region	Amazon SNS topic ARN
US East (N. Virginia) - `us-east-1`	`arn:aws:sns:us-east-1:242987662583:GuardDutyAnnouncements`
US East (Ohio) - `us-east-2`	`arn:aws:sns:us-east-2:118283430703:GuardDutyAnnouncements`

AWS Region	Amazon SNS topic ARN
US West (N. California) - `us-west-1`	`arn:aws:sns:us-west-1:144182107116:GuardDutyAnnouncements`
US West (Oregon) - `us-west-2`	`arn:aws:sns:us-west-2:934957504740:GuardDutyAnnouncements`
Canada (Central) - `ca-central-1`	`arn:aws:sns:ca-central-1:107430051933:GuardDutyAnnouncements`
Canada West (Calgary) - `ca-west-1`	`arn:aws:sns:ca-west-1:440427180217:GuardDutyAnnouncements`
Europe (Stockholm) - `eu-north-1`	`arn:aws:sns:eu-north-1:973841112453:GuardDutyAnnouncements`
Europe (Ireland) - `eu-west-1`	`arn:aws:sns:eu-west-1:965013871422:GuardDutyAnnouncements`
Europe (London) - `eu-west-2`	`arn:aws:sns:eu-west-2:506403581195:GuardDutyAnnouncements`

AWS Region	Amazon SNS topic ARN
Europe (Paris) - eu-west-3	arn:aws:sns:eu-west-3:436163563069:GuardDutyAnnouncements
Europe (Frankfurt) - eu-central-1	arn:aws:sns:eu-central-1:378365507264:GuardDutyAnnouncements
Europe (Zurich) - eu-central-2	arn:aws:sns:eu-central-2:383009515534:GuardDutyAnnouncements
Asia Pacific (Hong Kong) - ap-east-1	arn:aws:sns:ap-east-1:646602203151:GuardDutyAnnouncements
Asia Pacific (Tokyo) - ap-northeast-1	arn:aws:sns:ap-northeast-1:741172661024:GuardDutyAnnouncements
Asia Pacific (Seoul) - ap-northeast-2	arn:aws:sns:ap-northeast-2:464168911255:GuardDutyAnnouncements
Asia Pacific (Singapore) - ap-southeast-1	arn:aws:sns:ap-southeast-1:476419727788:GuardDutyAnnouncements

AWS Region	Amazon SNS topic ARN
Asia Pacific (Sydney) - ap-southeast-2	arn:aws:sns:ap-southeast-2:457615622431:GuardDutyAnnouncements
Asia Pacific (Mumbai) - ap-south-1	arn:aws:sns:ap-south-1:926826061926:GuardDutyAnnouncements
South America (São Paulo) - sa-east-1	arn:aws:sns:sa-east-1:955633302743:GuardDutyAnnouncements
AWS GovCloud (US-West) - us-gov-west-1	arn:aws-us-gov:sns:us-gov-west-1:430639793359:GuardDutyAnnouncements
China (Beijing) - cn-north-1	arn:aws-cn:sns:cn-north-1:002991280229:GuardDutyAnnouncements
China (Ningxia) - cn-northwest-1	arn:aws-cn:sns:cn-northwest-1:003033775354:GuardDutyAnnouncements
Middle East (Bahrain) - me-south-1	arn:aws:sns:me-south-1:552740612889:GuardDutyAnnouncements

AWS Region	Amazon SNS topic ARN
Middle East (UAE) - `me-central-1`	`arn:aws:sns:me-central-1:030935290150:GuardDutyAnnouncements`
Europe (Milan) - `eu-south-1`	`arn:aws:sns:eu-south-1:188461706213:GuardDutyAnnouncements`
Europe (Spain) - `eu-south-2`	`arn:aws:sns:eu-south-2:445632894446:GuardDutyAnnouncements`
AWS GovCloud (US-East) - `us-gov-east-1`	`arn:aws:sns:us-gov-east-1:143972945659:GuardDutyAnnouncements`
Asia Pacific (Osaka) - `ap-northeast-3`	`arn:aws:sns:ap-northeast-3:129086577509:GuardDutyAnnouncements`
Asia Pacific (Jakarta) - `ap-southeast-3`	`arn:aws:sns:ap-southeast-3:225965583551:GuardDutyAnnouncements`
Asia Pacific (Hyderabad) - `ap-south-2`	`arn:aws:sns:ap-south-2:595653072700:GuardDutyAnnouncements`

AWS Region	Amazon SNS topic ARN
Asia Pacific (Melbourne) - `ap-southeast-4`	`arn:aws:sns:ap-southeast-4:529900636122:GuardDutyAnnouncements`
Asia Pacific (Malaysia) - `ap-southeast-5`	`arn:aws:sns:ap-southeast-5:343218181797:GuardDutyAnnouncements`
Israel (Tel Aviv) - `il-central-1`	`arn:aws:sns:il-central-1:847886274986:GuardDutyAnnouncements`
Asia Pacific (Thailand) - `ap-southeast-7`	`arn:aws:sns:ap-southeast-7:863518448376:GuardDutyAnnouncements`
Mexico (Central) - `mx-central-1`	`arn:aws:sns:mx-central-1:060795916546:GuardDutyAnnouncements`
Asia Pacific (Taipei) - `ap-east-2`	`arn:aws:sns:ap-east-2:604225987917:GuardDutyAnnouncements`

To subscribe to the GuardDuty update notification email in the AWS Management Console

1. Open the Amazon SNS console at https://console.aws.amazon.com/sns/v3/home.
2. In the Region list, choose the same Region as the topic ARN to which to subscribe. This example uses the `us-west-2` Region.

3. In the left navigation pane, choose **Subscriptions**, **Create subscription**.
4. In the **Create Subscription** dialog box, for **Topic ARN**, paste the topic ARN: `arn:aws:sns:us-west-2:934957504740:GuardDutyAnnouncements`.
5. For **Protocol**, choose **Email**. For **Endpoint**, type an email address that you can use to receive the notification.
6. Choose **Create subscription**.
7. In your email application, open the message from AWS Notifications and open the link to confirm your subscription.

   Your web browser displays a confirmation response from Amazon SNS.

To subscribe to the GuardDuty update notification email with the AWS CLI

1. Run the following command with the AWS CLI:

```
 aws  sns --region us-west-2 subscribe --topic-arn arn:aws:sns:us-
west-2:934957504740:GuardDutyAnnouncements --protocol email --notification-
endpoint your_email@your_domain.com
```

2. In your email application, open the message from AWS Notifications and open the link to confirm your subscription.

   Your web browser displays a confirmation response from Amazon SNS.

Amazon SNS message format

An example GuardDuty general notification message:

```
{
    "Type" : "Notification",
    "MessageId" : "9101dc6b-726f-4df0-8646-ec2f94e674bc",
    "TopicArn" : "arn:aws:sns:us-west-2:934957504740:GuardDutyAnnouncements",
    "Message" : "{\"version\":\"1\",\"type\":\"GENERAL\",\"message\":[{\"title\":
\"Updated AmazonGuardDutyFullAccess_v2 policy\",\"body\":\"Added permission that
 allows you to pass an IAM role to GuardDuty when you enable Malware Protection for
 S3.\",\"links\":[\"https://docs.aws.amazon.com//guardduty/latest/ug/security-iam-
awsmanpol.html#security-iam-awsmanpol-AmazonGuardDutyFullAccess-v2\"]}]}",
    "Timestamp" : "2018-03-09T00:25:43.483Z",
```

```
    "SignatureVersion" : "1",
    "Signature" : "XWox8GDGLRiCgDOXlo/
fG9Lu/88P8S0FL6M6oQYOmUFzkucuhoblsdea3BjqdCHcWR7qdhMPQnLpN7y9iBrWVUqdAGJrukAI8athvAS
+4AQD/V/QjrhsEnlj+GaiW
+ozAu006X6GopOzFGnCtPMROjCMrMonjz7Hpv/8KRuMZR3pyQYm5d4wWB7xBPYhUMuLoZ1V8YFs55FMtgQV/
YLhSYuEu0BP1GMtLQauxDkscOtPP/vjhGQLFx1Q9LTadcQiRHtNIBxWL87PSI
+BVvkin6AL7PhksvdQ7FAgHfXsit+6p8GyOvKCqaeBG7HZhR1AbpyVka7JSNRO/6ssyrlj1g==",
    "SigningCertURL" : "https://sns.us-west-2.amazonaws.com/
SimpleNotificationService-433026a4050d206028891664da859041.pem",
    "UnsubscribeURL" : "https://sns.us-west-2.amazonaws.com/?
Action=Unsubscribe&SubscriptionArn=arn:aws:sns:us-
west-2:934957504740:GuardDutyAnnouncements:9225ed2b-7228-4665-8a01-c8a5db6859f4"
}
```

The parsed Message value (with escaped quotes removed) is shown below:

```
{
        "version": "1",
        "type": "GENERAL",
        "message": [
            {
                "title": "Updated AmazonGuardDutyFullAccess policy",
                "body": "Added permission that allows you to pass an IAM role to
 GuardDuty when you enable Malware Protection for S3.",
                "links": [
                    "https://docs.aws.amazon.com//guardduty/latest/ug/security-iam-
awsmanpol.html#security-iam-awsmanpol-AmazonGuardDutyFullAccess-v2"
                ]
             }
        ]
}
```

An example GuardDuty update notification message about new findings is shown below:

```
{
    "Type" : "Notification",
    "MessageId" : "9101dc6b-726f-4df0-8646-ec2f94e674bc",
    "TopicArn" : "arn:aws:sns:us-west-2:934957504740:GuardDutyAnnouncements",
    "Message" : "{\"version\":\"1\",\"type\":\"NEW_FINDINGS\",\"findingDetails
\":[{\"link\":\"https://docs.aws.amazon.com//guardduty/latest/ug/
guardduty_unauthorized.html\",\"findingType\":\"UnauthorizedAccess:EC2/TorClient\",
\"findingDescription\":\"This finding informs you that an EC2 instance in your AWS
 environment is making connections to a Tor Guard or an Authority node. Tor is software
```

```
 for enabling anonymous communication. Tor Guards and Authority nodes act as initial
 gateways into a Tor network. This traffic can indicate that this EC2 instance is
 acting as a client on a Tor network. A common use for a Tor client is to circumvent
 network monitoring and filter for access to unauthorized or illicit content. Tor
 clients can also generate nefarious Internet traffic, including attacking SSH servers.
 This activity can indicate that your EC2 instance is compromised.\"}]}",
    "Timestamp" : "2018-03-09T00:25:43.483Z",
    "SignatureVersion" : "1",
    "Signature" : "XWox8GDGLRiCgDOXlo/
fG9Lu/88P8S0FL6M6oQYOmUFzkucuhoblsdea3BjqdCHcWR7qdhMPQnLpN7y9iBrWVUqdAGJrukAI8athvAS
+4AQD/V/QjrhsEnlj+GaiW
+ozAu006X6GopOzFGnCtPMROjCMrMonjz7Hpv/8KRuMZR3pyQYm5d4wWB7xBPYhUMuLoZ1V8YFs55FMtgQV/
YLhSYuEu0BP1GMtLQauxDkscOtPP/vjhGQLFx1Q9LTadcQiRHtNIBxWL87PSI
+BVvkin6AL7PhksvdQ7FAgHfXsit+6p8GyOvKCqaeBG7HZhR1AbpyVka7JSNRO/6ssyrlj1g==",
    "SigningCertURL" : "https://sns.us-west-2.amazonaws.com/
SimpleNotificationService-433026a4050d206028891664da859041.pem",
    "UnsubscribeURL" : "https://sns.us-west-2.amazonaws.com/?
Action=Unsubscribe&SubscriptionArn=arn:aws:sns:us-
west-2:934957504740:GuardDutyAnnouncements:9225ed2b-7228-4665-8a01-c8a5db6859f4"
}
```


The parsed Message value (with escaped quotes removed) is shown below:


```
{
    "version": "1",
    "type": "NEW_FINDINGS",
    "findingDetails": [{
        "link": "https://docs.aws.amazon.com//guardduty/latest/ug/
guardduty_unauthorized.html",
        "findingType": "UnauthorizedAccess:EC2/TorClient",
        "findingDescription": "This finding informs you that an EC2 instance in your
 AWS environment is making connections to a Tor Guard or an Authority node. Tor is
 software for enabling anonymous communication. Tor Guards and Authority nodes act as
 initial gateways into a Tor network. This traffic can indicate that this EC2 instance
 is acting as a client on a Tor network. A common use for a Tor client is to circumvent
 network monitoring and filter for access to unauthorized or illicit content. Tor
 clients can also generate nefarious Internet traffic, including attacking SSH servers.
 This activity can indicate that your EC2 instance is compromised."
    }]
}
```


An example GuardDuty update notification message about GuardDuty functionality updates is shown below:

```
{
    "Type" : "Notification",
    "MessageId" : "9101dc6b-726f-4df0-8646-ec2f94e674bc",
    "TopicArn" : "arn:aws:sns:us-west-2:934957504740:GuardDutyAnnouncements",
    "Message" : "{\"version\":\"1\",\"type\":\"NEW_FEATURES\",\"featureDetails
\":[{\"featureDescription\":\"Customers with high-volumes of global CloudTrail
 events should see a net positive impact on their GuardDuty costs.\",\"featureLink
\":\"https://docs.aws.amazon.com//guardduty/latest/ug/guardduty_data-
sources.html#guardduty_controlplane\"}]}",
    "Timestamp" : "2018-03-09T00:25:43.483Z",
    "SignatureVersion" : "1",
    "Signature" : "XWox8GDGLRiCgDOXlo/
fG9Lu/88P8S0FL6M6oQYOmUFzkucuhoblsdea3BjqdCHcWR7qdhMPQnLpN7y9iBrWVUqdAGJrukAI8athvAS
+4AQD/V/QjrhsEnlj+GaiW
+ozAu006X6GopOzFGnCtPMROjCMrMonjz7Hpv/8KRuMZR3pyQYm5d4wWB7xBPYhUMuLoZ1V8YFs55FMtgQV/
YLhSYuEu0BP1GMtLQauxDkscOtPP/vjhGQLFx1Q9LTadcQiRHtNIBxWL87PSI
+BVvkin6AL7PhksvdQ7FAgHfXsit+6p8GyOvKCqaeBG7HZhR1AbpyVka7JSNRO/6ssyrlj1g==",
    "SigningCertURL" : "https://sns.us-west-2.amazonaws.com/
SimpleNotificationService-433026a4050d206028891664da859041.pem",
    "UnsubscribeURL" : "https://sns.us-west-2.amazonaws.com/?
Action=Unsubscribe&SubscriptionArn=arn:aws:sns:us-
west-2:934957504740:GuardDutyAnnouncements:9225ed2b-7228-4665-8a01-c8a5db6859f4"
}
```

The parsed Message value (with escaped quotes removed) is shown below:

```
{
    "version": "1",
    "type": "NEW_FEATURES",
    "featureDetails": [{
        "featureDescription": "Customers with high-volumes of global CloudTrail events
 should see a net positive impact on their GuardDuty costs.",
        "featureLink": "https://docs.aws.amazon.com//guardduty/latest/ug/
guardduty_data-sources.html#guardduty_controlplane"
    }]
}
```

An example GuardDuty update notification message about updated findings is shown below:

```
{
    "Type": "Notification",
    "MessageId": "9101dc6b-726f-4df0-8646-ec2f94e674bc",
```

```
    "TopicArn": "arn:aws:sns:us-west-2:934957504740:GuardDutyAnnouncements",
    "Message": "{\"version\":\"1\",\"type\":\"UPDATED_FINDINGS\",
\"findingDetails\":[{\"link\":\"https://docs.aws.amazon.com//guardduty/latest/ug/
guardduty_unauthorized.html\",\"findingType\":\"UnauthorizedAccess:EC2/TorClient\",
\"description\":\"Increased severity value from 5 to 8.\"}]}",
    "Timestamp": "2018-03-09T00:25:43.483Z",
    "SignatureVersion": "1",
    "Signature": "XWox8GDGLRiCgDOXlo/
fG9Lu/88P8S0FL6M6oQYOmUFzkucuhoblsdea3BjqdCHcWR7qdhMPQnLpN7y9iBrWVUqdAGJrukAI8athvAS
+4AQD/V/QjrhsEnlj+GaiW
+ozAu006X6GopOzFGnCtPMROjCMrMonjz7Hpv/8KRuMZR3pyQYm5d4wWB7xBPYhUMuLoZ1V8YFs55FMtgQV/
YLhSYuEu0BP1GMtLQauxDkscOtPP/vjhGQLFx1Q9LTadcQiRHtNIBxWL87PSI
+BVvkin6AL7PhksvdQ7FAgHfXsit+6p8GyOvKCqaeBG7HZhR1AbpyVka7JSNRO/6ssyrlj1g==",
    "SigningCertURL": "https://sns.us-west-2.amazonaws.com/
SimpleNotificationService-433026a4050d206028891664da859041.pem",
    "UnsubscribeURL": "https://sns.us-west-2.amazonaws.com/?
Action=Unsubscribe&SubscriptionArn=arn:aws:sns:us-
west-2:934957504740:GuardDutyAnnouncements:9225ed2b-7228-4665-8a01-c8a5db6859f4"
}
```


The parsed Message value (with escaped quotes removed) is shown below:


```
{
    "version": "1",
    "type": "UPDATED_FINDINGS",
    "findingDetails": [{
        "link": "https://docs.aws.amazon.com//guardduty/latest/ug/
guardduty_unauthorized.html",
        "findingType": "UnauthorizedAccess:EC2/TorClient",
        "description": "Increased severity value from 5 to 8."
    }]
}
```

Amazon GuardDuty quotas

Your AWS account has default quotas, formerly referred to as limits, for each AWS service. Unless otherwise noted, each quota is Region-specific. You can request increases for some quotas, and other quotas can't be increased.

To view the quotas for GuardDuty, open the Service Quotas console. In the navigation pane, choose **AWS services** and select **Amazon GuardDuty**.

To request a quota increase, see Requesting a quota increase in the *Service Quotas User Guide*.

Your AWS account has the following quotas for Amazon GuardDuty per Region.

> **Note**
>
> - For quotas specific to GuardDuty Malware Protection for EC2, see Quotas in Malware Protection for EC2.
> - For quotas specific to Malware Protection for S3, see Quotas in Malware Protection for S3.

GuardDuty quotas per Region

Resource	Default	Comments
Detectors	1	The maximum number of detector resources that you can create per AWS account per Region. You can't request a quota increase.
Filters	100	The maximum number of saved filters per AWS account per Region.

Resource	Default	Comments
		You can't request a quota increase.
Finding retention period	90 days	The maximum number of days a finding is retained. You can't request a quota increase.
IP addresses and CIDR ranges per trusted IP address list	2,000	The maximum number of IP addresses and CIDR ranges that you can include in a single trusted IP address list. You can't request a quota increase.
IP addresses and CIDR ranges per threat IP address list	250,000	The maximum number of IP address and CIDR ranges that you can include in a threat IP address list. You can't request a quota increase.

Resource	Default	Comments
Entities per threat entity list	1,000	The maximum number of entities that you can include in a single threat entity list. You can't request a quota increase.
Entities per trusted entity list	1,000	The maximum number of entities that you can include in a trusted entity list. You can't request a quota increase.
Maximum file size	35 MB	The maximum file size for the file used to upload an entity list or an IP address list. You can't request a quota increase.
Member accounts (by invitation)	5000	The maximum number of member accounts associated with a administrator account. You can't request a quota increase.

Resource	Default	Comments
Member accounts	50,000	The maximum number of member accounts associate d with a administr ator account through AWS Organizat ions. This includes member accounts that are added to the organization by invitation. This default value depends on your current quota for member accounts in AWS Organizat ions. The number of member accounts in GuardDuty that are added through AWS Organizations can't exceed the number of member accounts in your organizat ion. For informati on about number of AWS accounts in an organization, see Maximum and minimum values in the *AWS Organizat ions User Guide*.

Resource	Default	Comments
Threat intel sets (IP address list)	6	The maximum number of threat IP address list that you can add per AWS account per Region. You can't request a quota increase.
Trusted IP sets (IP address list)	1	The maximum number of trusted IP address list that can be uploaded and activated per AWS account per Region. You can't request a quota increase.
Threat entity lists	6	The maximum number of threat entity lists that you can add per AWS account per Region. You can't request a quota increase.

Resource	Default	Comments
Trusted entity list	1	The maximum number of trusted entity list that can be uploaded and activated per AWS account per Region. You can't request a quota increase.

Troubleshooting Amazon GuardDuty

When you receive issues related to performing an action specific to GuardDuty, consult the topics in this section.

Topics

- Exporting findings to Amazon S3 - access error
- Malware Protection for EC2 issues
- Runtime Monitoring issues
- Other troubleshooting issues

Exporting findings to Amazon S3 - access error

When you export GuardDuty findings to an Amazon S3 bucket (publishing destination), if GuardDuty is unable to access this publishing destination, then you may get an access error.

After you configure settings to export findings, if GuardDuty is unable to export findings, it displays an error message on the **Settings** page in the GuardDuty console. This can potentially happen when GuardDuty can no longer access the target resource. For example, if your Amazon S3 bucket was deleted or the permission to access the bucket was modified. This can also potentially happen when GuardDuty can no longer access the AWS KMS key that was used to encrypt the data in your Amazon S3 bucket. When GuardDuty is unable to export, it sends a notification to the email associated with the account to provide information about this issue.

How to resolve the access error?

To resolve the issue, make sure that the corresponding resources exist and GuardDuty has the permissions to access the needed resources.

For more information, see Exporting generated findings to Amazon S3.

What happens when you don't resolve this error?

If you don't resolve the issue before the 90-day finding retention period completes in GuardDuty, your findings will not get exported. GuardDuty will disable finding export settings for this account in the specific Region.

To start exporting the findings again, update the configuration settings in the specific Region.

Malware Protection for EC2 issues

This section lists the errors that you may experience when setting up or using Malware Protection for EC2.

Missing required AWS Organizations management permission when enabling GuardDuty-initiated malware scan

When you want to manage multiple accounts by using AWS Organizations and you get this error – `The request failed because you do not have required AWS Organization master permission.`, then you're missing the permission to enable GuardDuty-initiated malware scan for multiple accounts in your organization.

For information about providing permissions to the management account, see Establishing trusted access to enable GuardDuty-initiated malware scan.

I am initiating an On-demand malware scan but it results in a missing required permissions error.

If you receive an error suggesting that you do not have the required permissions to start an On-demand malware scan on an Amazon EC2 instance, verify that you've attached the AWS managed policy: AmazonGuardDutyFullAccess_v2 (recommended) policy to your IAM role.

If you're a member of an AWS organization and still receive the same error, connect with your management account. For more information, see AWS Organizations SCP – Denied access.

I receive an `iam:GetRole` error while working with Malware Protection for EC2.

If you receive this error – `Unable to get role: AWSServiceRoleForAmazonGuardDutyMalwareProtection`, it means that you're missing the permission to either enable GuardDuty-initiated malware scan or use On-demand malware scan. Verify that you've attached the AWS managed policy: AmazonGuardDutyFullAccess_v2 (recommended) policy to your IAM role.

I am a GuardDuty administrator account who needs to enable GuardDuty-initiated malware scan but doesn't use AWS managed policy: AmazonGuardDutyFullAccess to manage GuardDuty.

- Configure the IAM role that you use with GuardDuty to have the required permissions to enable GuardDuty-initiated malware scan. For more information on the required permissions, see Creating a service-linked role for Malware Protection for EC2.
- Attach the AWS managed policy: AmazonGuardDutyFullAccess_v2 (recommended) to your IAM role. This will help you enable GuardDuty-initiated malware scan for the member accounts.

Runtime Monitoring issues

This section lists the errors that you may experience when setting up or using Runtime Monitoring.

Runtime coverage issues

When the runtime coverage of your protected resources become **Unhealthy**, the GuardDuty console provides the exact issue type. After you have the issue type, use the following documents to view the troubleshooting steps for each supported resource type:

- Troubleshooting Amazon EC2 runtime coverage issues
- Troubleshooting Amazon ECS-Fargate runtime coverage issues
- Troubleshooting Amazon EKS runtime coverage issues

Troubleshooting out of memory error in Runtime Monitoring (Amazon EC2 support only)

This section provides the troubleshooting steps when you experience out of memory error based on the CPU and memory limit to deploy the GuardDuty security agent manually.

If `systemd` terminates the GuardDuty agent because of the `out-of-memory` issue and you evaluate that providing more memory to the GuardDuty agent is reasonable, you can update the limit.

1. With the root permission, open `/lib/systemd/system/amazon-guardduty-agent.service`.

2. Find `MemoryLimit` and `MemoryMax`, and update both the values.

```
MemoryLimit=256MB
MemoryMax=256MB
```

3. After updating the values, restart the GuardDuty agent by using the following command:

```
sudo systemctl daemon-reload
sudo systemctl restart amazon-guardduty-agent
```

4. Run the following command to view the status:

```
sudo systemctl status amazon-guardduty-agent
```

   The expected output will show the new memory limit:

```
Main PID: 2540 (amazon-guardduty)
Tasks: 16
Memory: 21.9M (limit: 256.0M)
```

My AWS Step Functions workflow is failing unexpectedly

If the GuardDuty container contributed to the workflow failure, see Troubleshooting Amazon ECS-Fargate runtime coverage issues. If the issue persists, then to prevent the workflow failure because of the GuardDuty container, perform **one** of the following steps:

- Add the `GuardDutyManaged`:`false` tag to associated Amazon ECS cluster.
- Disable the automated agent configuration for AWS Fargate (ECS only) at the account level. Add the inclusion tag `GuardDutyManaged`:`true` to the associated Amazon ECS cluster that you want to continue monitoring with the GuardDuty automated agent.

Other troubleshooting issues

If you don't find a scenario suitable to your issue, view the following troubleshooting options:

- For general IAM issues when you access the https://console.aws.amazon.com/guardduty/, see Troubleshooting Amazon GuardDuty identity and access.

- For authentication and authorization issues when you access AWS AWS Console Home, see Troubleshooting IAM.

Amazon GuardDuty Regions and endpoints

To view the AWS Regions where Amazon GuardDuty is available, see Amazon GuardDuty endpoints in the *Amazon Web Services General Reference*.

We recommend that you enable GuardDuty in all supported AWS Regions. This enables GuardDuty to generate findings about unauthorized or unusual activity even in Regions that you are not actively using. This also allows GuardDuty to monitor AWS CloudTrail events for the supported AWS Regions, its ability to detect activity that involves global services is reduced.

Region-specific feature availability

A list of regional differences to specify the availability of GuardDuty features.

ListFindings and GetFindingsStatistics APIs

The GetFindingsStatistics and ListFindings APIs have a temporary `consoleOnly` flag. When you use any or both of these APIs, the `consoleOnly` flag means that the API can fetch results to a maximum limit of 1000.

Malware Protection for EC2

GuardDuty supports the Malware Protection for EC2 feature in the AWS Dedicated Local Zones.

RDS Protection

RDS Protection is not supported in Asia Pacific (Taipei) (`ap-east-2`) Region.

General API support

The following APIs in the Amazon GuardDuty API Reference may have regional differences because of the unavailability of some of the data sources or features in previously specified AWS Regions:

- CreateDetector
- UpdateDetector
- UpdateMemberDetectors
- UpdateOrganizationConfiguration
- GetDetector
- GetMemberDetectors

- DescribeOrganizationConfiguration

Amazon EC2 finding types – DefenseEvasion:EC2/UnusualDoHActivity and DefenseEvasion:EC2/UnusualDoTActivity

The following table shows the AWS Regions where GuardDuty is available but these two Amazon EC2 finding types are not yet supported.

AWS Region	Region code
Asia Pacific (Seoul)	ap-northeast-2
Asia Pacific (Osaka)	ap-northeast-3
Asia Pacific (Jakarta)	ap-southeast-3

AWS GovCloud (US) Regions

For latest information, see Amazon GuardDuty in the *AWS GovCloud (US) User Guide*.

China Regions

For latest information, see Feature availability and implementation differences.

GuardDuty legacy actions and parameters

Amazon GuardDuty has deprecated the some of the API actions and parameters but still supports them. The best practice is to use the new API actions and parameters that replace the legacy options. The following table compares the legacy and new actions and parameters.

Legacy actions/ parameters	New actions/parameters	Comparison
DisassociateFromMasterAccount	DisassociateFromAdministratorAccount	With the same implementation in both the actions, GuardDuty uses the term `Administrator` in `DisassociateFromAdministratorAccount` .
`autoEnable` parameter in DescribeOrganizationConfiguration and UpdateOrganizationConfiguration	`autoEnableOrganizationMembers`	With `autoEnableOrganizationMembers` , the GuardDuty administrator account can audit and enforce GuardDuty for all member accounts to either of the values. Using the APIs, it may take up to 24 hours to update the configuration for all the member accounts. For more information about the possible values of the `autoEnableOrganizationMembers` field, see autoEnableOrganizationMembers
`dataSources` parameter in the APIs listed in GuardDuty API changes in March 2023.	`features`	Starting March 2023, you can configure GuardDuty Malware Protection for EC2 and the new GuardDuty protection plans using `features`. The protection plans launched prior to March 2023, including Malware Protection for

Legacy actions/ parameters	New actions/parameters	Comparison
		EC2 still support configuration using `dataSources` . If you use APIs to configure a protection plan, each API request can either include `dataSources` or `features`, not both.

Document history for Amazon GuardDuty

The following table describes important changes to the documentation since the last release of the *Amazon GuardDuty User Guide*. For notification about updates to this documentation, you can subscribe to an RSS feed.

Change	Description	Date
Updated functionality - Malware Protection for S3	GuardDuty Malware Protection for S3 increases the **Extracted archive files** default quota from 1,000 to 10,000 files. For more information, see Quotas in Malware Protection for S3.	September 3, 2025
Updated functionality - Runtime Monitoring	GuardDuty Runtime Monitoring releases the new security agent version 1.11 for Amazon EKS resources. For more information about the new agent version and a list of additional resources to update your security agent, see GuardDuty security agent release versions.	August 29, 2025
Updated functionality - Working with lists	GuardDuty introduces custom trusted and threat entity lists that support both IP addresses and domain names. GuardDuty continues supporting IP address list and recommends using the entity list for custom threat detection. For more information, see Customizing threat	August 15, 2025

	detection with entity lists and IP address lists.	
Updated functionality - Runtime Monitoring	GuardDuty Runtime Monitoring releases the new security agent version 1.8.0 for both Amazon EC2 and Amazon ECS-AWS Fargate resources. For more information about new agent versions and a list of additional resources to update your security agent, see GuardDuty security agent release versions.	August 12, 2025
Support for Asia Pacific (Taipei) Region	Amazon GuardDuty is now available in the Asia Pacific (Taipei) (`ap-east-2`) Region. To enable GuardDuty in this Region, see Getting started. You can receive notifications about updates to the GuardDuty features and threat detections by Subscribing to Amazon SNS GuardDuty announcements in this Region.	July 31, 2025
Updated functionality - Malware Protection for S3	Malware Protection for S3 now supports scanning objects up to 100 GB, increased from 5 GB. This includes both individual objects and extracted archive files. For more information, see Malware Protection for S3 quotas.	July 23, 2025

Updated functionality - Added expected bucket owner in trust IP and threat lists	GuardDuty added the **Expected bucket owner** field for trusted IP and threat IP lists. In this optional field, you can specify an AWS account ID that GuardDuty will use to verify Amazon S3 bucket ownership. For more information, see Working with trusted IP lists and threat lists.	July 16, 2025
Updated functionality - Extended Threat Detection	GuardDuty Extended Threat Detection now expands support for Amazon EKS clusters by correlating multiple security signals across EKS audit logs, runtime behavior of processes, and AWS API activity. Enable EKS Protection, Runtime Monitoring (with EKS add-on), or both to maximize threat detection. To identify potential threats, GuardDuty introduces a new finding type – AttackSequence:EKS/CompromisedCluster. For more information, see Extended Threat Detection.	June 17, 2025

Updated functionality - Malware Protection for S3	Malware Protection for S3 doesn't support scanning archives with extremely high compression ratio. These files will be skipped during scanning and marked with a scan result of `UNSUPPORTED` . For more information, see S3 object potential scan status and result status.	June 13, 2025
Updated functionality - Malware Protection for EC2	Malware Protection for EC2 has added support for scanning majority of instances with `productCode` as `marketplace` . This applies to both GuardDuty-initiated malware scan and On-demand malware scans. For more information, see Reasons for skipping resource during malware scan.	June 13, 2025
New AmazonGuardDutyFullAccess_v2 policy	Added a new AmazonGuardDutyFullAccess_v2 policy with permissions to enhance security by restricting administrative actions to GuardDuty service principals. For information about this recommended policy, see AWS managed policy: AmazonGuardDutyFullAccess_v2.	June 4, 2025

Expanded Region support for RDS Protection	GuardDuty RDS Protection is now available in Mexico (Central), Asia Pacific (Thailand), and Asia Pacific (Malaysia) Regions. RDS Protection helps you detect potentially suspicious login behavior in supported Aurora MySQL, Aurora PostgreSQL (including Limitless Database) , and RDS for PostgreSQL. In the event of threat detection, GuardDuty generates an RDS Protection finding. For more information about supported databases and enabling this protection plan in the newly supported Regions, see RDS Protection.	June 4, 2025
Updated functionality - Runtime Monitoring	GuardDuty Runtime Monitoring releases new security agent version 1.7.1 for Amazon EC2 resources. For more information about new agent versions and a list of additional resources to update your security agent, see GuardDuty security agent release versions.	June 3, 2025

Support for Extended Threat Detection	GuardDuty Extended Threat Detection is now available in Asia Pacific (Thailand) (ap-southeast-7). With no activation needed, it detects multi-stage attacks that span data sources, multiple types of AWS resources, and time, within an AWS account. When potential threats are detected, it generates an attack sequence finding. For more information, see Extended Threat Detection.	May 12, 2025
Support for Mexico (Central) Region	Amazon GuardDuty is now available in the Mexico (Central) (`mx-central-1`) Region. To enable GuardDuty in this Region, see Getting started. You can receive notifications about updates to the GuardDuty features and threat detections by Subscribing to Amazon SNS GuardDuty announcements in this Region.	May 7, 2025

Updated functionality - Runtime Monitoring	GuardDuty Runtime Monitoring releases new security agent version 1.10.0 for Amazon EKS resources. For more information about new agent versions and a list of additional resources to update your security agent, see GuardDuty security agent release versions.	April 4, 2025
Updated functionality - Runtime Monitoring	GuardDuty Runtime Monitoring releases new security agent version 1.7.0 for Amazon ECS-Fargate resources. For more information about new agent versions and a list of additional resources to update your security agent, see GuardDuty security agent release versions.	April 4, 2025
Updated functionality - Runtime Monitoring	GuardDuty Runtime Monitoring releases new security agent version 1.7.0 for Amazon EC2 resources. For more information about new agent versions and a list of additional resources to update your security agent, see GuardDuty security agent release versions.	April 3, 2025

Support for Asia Pacific (Thailand) Region	Amazon GuardDuty is now available in the Asia Pacific (Thailand) Region. For information about which features are supported in this Region, see Region-specific feature availability. To enable GuardDuty in this Region, see Getting started. You can receive notifications about updates to the GuardDuty features and threat detection s by Subscribing to Amazon SNS GuardDuty announcem ents.	April 1, 2025
Updated functionality	The Summary dashboard now shows insights based on all the generated security findings, removing the previous 5,000 findings constraint. For informati on about these insights, see GuardDuty Summary dashboard.	March 17, 2025
Updated functionality - Runtime Monitoring	GuardDuty Runtime Monitorin g releases new security agent version 1.9.0 for Amazon EKS resources. For more information about new agent versions and a list of additional resources to update your security agent, see GuardDuty security agent release versions.	March 2, 2025

Updated functionality - Runtime Monitoring	GuardDuty Runtime Monitoring has added a new coverage issue type (Agent Not Provisioned) for Amazon EC2 resources. For information about troubleshooting this issue, see Troubleshooting Amazon EC2 runtime coverage issues.	February 21, 2025
Updated functionality - Runtime Monitoring	GuardDuty Runtime Monitoring releases new security agents for Amazon EC2 and Amazon ECS-Fargate resources. For more information about new agent versions and a list of additional resources to update your security agents, see GuardDuty security agent release versions.	February 6, 2025
GuardDuty support in existing Asia Pacific (Malaysia) Region	GuardDuty Extended Threat Detection is now available in the Asia Pacific (Malaysia) Region. For more information, see Extended Threat Detection.	January 28, 2025

Support for Asia Pacific (Malaysia) Region	Amazon GuardDuty is now available in the Asia Pacific (Malaysia) Region. For information about which features are supported in this Region, see Region-specific feature availability. To enable GuardDuty in this Region, see Getting started. You can receive notifications about updates to the GuardDuty features and threat detections by Subscribing to Amazon SNS GuardDuty announcements.	January 16, 2025
Updated functionality - Runtime Monitoring	GuardDuty Runtime Monitoring has updated extra information and troubleshooting steps for Amazon ECS-Fargate coverage issues associated with *Agent not provisioned*. For more information about *Agent not provisioned* issue type, see Troubleshooting Amazon ECS-Fargate runtime coverage issues.	January 8, 2025

New finding type - Policy:IAMUser/ShortTermRootCredentialUsage	GuardDuty introduces a new finding type that alerts you when restricted user credentials, created for the listed AWS accounts in your environment, are being used to make requests to AWS services. For more information, see Policy:IAMUser/ShortTermRootCredentialUsage.	January 8, 2025
New feature - GuardDuty Extended Threat Detection	GuardDuty announces Extended Threat Detection to detects multi-stage attack sequences that span GuardDuty foundational data sources and AWS resources in your AWS account, over a specific time period. At no additional cost, this capability is automatically enabled for all accounts that have enabled GuardDuty. This feature announces two new GuardDuty finding types, called Attack sequence finding types. For more information, see Extended Threat Detection.	December 1, 2024

Enhanced cross-service functionality - Runtime Monitoring and Malware Protection for EC2	Impact of new Amazon Elastic Kubernetes Service (Amazon EKS) features on Amazon GuardDuty features: • Amazon EKS Auto Mode – Both Runtime Monitoring for Amazon EKS and Malware Protection for EC2 support this. • Amazon EKS Hybrid Nodes – Both Runtime Monitoring for Amazon EKS and Malware Protection for EC2 don't support this. For more information, see How Runtime Monitoring works with Amazon EKS clusters and Malware Protection for EC2.	December 1, 2024

Updated functionality in Runtime Monitoring - Amazon EKS	Runtime Monitoring released a new agent version 1.8.1 (*v1.8.1-eks-build.2*) for Amazon EKS resources. With this new agent version, GuardDuty extends Runtime Monitoring support for Amazon EKS resources that run on RedHat, CentOS, and Fedora. For more information, see Validating architectural requirements. For informati on about release notes, see GuardDuty security agent for Amazon EKS resources.	November 23, 2024
Updated functionality in Runtime Monitoring - Amazon EC2	Runtime Monitoring released a new agent version 1.5.0 for Amazon EC2 resources. With this new agent version, GuardDuty extends Runtime Monitoring support for Amazon EC2 resources that run on RedHat, CentOS, and Fedora. For more information, see Validating architectural requirements. For informati on about release notes, see GuardDuty security agent for Amazon EC2 resources.	November 20, 2024

Updated functionality in Runtime Monitoring - Amazon ECS-Fargate	Runtime Monitoring released a new agent version 1.5.0 for Amazon ECS-Fargate resources. For more information about release notes, see GuardDuty security agent for AWS Fargate (Amazon ECS only).	November 14, 2024
Updated functionality in Malware Protection for EC2	GuardDuty Malware Protection for EC2 has added three Runtime Monitoring finding types to the list of Findings that invoke GuardDuty-initiated malware scan on Amazon EC2 instances. Accounts that have enabled Malware Protection for EC2 will observe GuardDuty-initiated malware scan when GuardDuty generates any of the following findings: • Execution:Runtime/MaliciousFileExecuted • Execution:Runtime/SuspiciousShellCreated • PrivilegeEscalation:Runtime/ElevationToRoot	November 7, 2024

Updated functionality in RDS Protection	GuardDuty RDS Protection adds the newly released Aurora PostgreSQL Limitless Database engine version `16.4-limitless` to the list of supported databases. For AWS accounts that have already enabled RDS Protection, GuardDuty will automatically start monitoring the login behavior for the Limitless Database. Accounts that have already consumed the 30-day free trial for RDS Protection will incur usage cost associated with Limitless Database, along with other supported databases that are monitored. For more information, see RDS Protection.	November 6, 2024
Region expansion - GuardDuty and AWS PrivateLink integration	GuardDuty now extends Region support for Amazon GuardDuty and interface VPC endpoints (AWS PrivateLink). Earlier, the Region support was available for US East (N. Virginia), Europe (Ireland), and Israel (Tel Aviv). This support is now extended to all the AWS Regions where GuardDuty is available. For more information on regional differences, see Region-specific feature availability.	November 6, 2024

Updated functionality in Runtime Monitoring - Amazon ECS-Fargate	Runtime Monitoring released a new agent version 1.4.1 for Amazon ECS-Fargate resources. For more informati on about release notes, see GuardDuty security agent for AWS Fargate (Amazon ECS only).	October 24, 2024
Added support for GuardDuty CloudFormation tag operations	GuardDuty now supports updating tag key and value, and stack-level tags. To do this, add `guardduty :tagResource` permissio n to the IAM role. For information about GuardDuty CloudFormation, see Amazon GuardDuty resource type reference in the *AWS CloudFormation User Guide*.	October 24, 2024
Updated functionality in GuardDuty Malware Protectio n for S3	When enabling malware protection for S3, you can choose a service role that has the necessary permissions to perform malware scan actions on your behalf. For more information about enabling Malware Protection for S3, see Configuring Malware Protection for S3 for your S3 bucket.	October 22, 2024

Updated functionality	GuardDuty enhances the UnauthorizedAccess:IAMUser/InstanceCredentialExfiltration.InsideAWS finding type to detect the use of Amazon EC2 instance AWS credentials from VPC endpoints (AWS PrivateLink) in AWS accounts that are not associated with the Amazon EC2 instance role. This new GuardDuty capability detects potential Amazon EC2 instance credential misuse and provides context of the remote AWS account using the exfiltrating session credentials. For more information about AWS service endpoints supported by this new detection, see Logging network activity events in the *AWS CloudTrail User Guide*.	October 21, 2024

Updated functionality - GuardDuty Runtime Monitoring	GuardDuty Runtime Monitoring added the following three finding types that notify you when suspicious commands are executed on an Amazon EC2 instance or container workload within your AWS environment: • Discovery:Runtime/SuspiciousCommand • Persistence:Runtime/SuspiciousCommand • PrivilegeEscalation:Runtime/SuspiciousCommand	October 10, 2024
New feature - Added support for VPC endpoints	GuardDuty is now integrated with AWS PrivateLink and supports VPC endpoints. For more information about the AWS PrivateLink integration, see Amazon GuardDuty and interface VPC endpoints (AWS PrivateLink).	September 17, 2024
Updated functionality in Runtime Monitoring - Amazon EKS	Runtime Monitoring released a new agent version 1.7.1 for Amazon EKS resources. For more information about release notes, see GuardDuty security agent for Amazon EKS.	September 13, 2024

Updated functionality in Malware Protection for S3	Malware Protection for S3 added a new field, `s3Throttled` , to the S3 object scan result Amazon EventBridge (EventBridge) schema. The `s3Throttled` field indicates whether or not there was a delay in uploading or retrieving storage from Amazon Simple Storage Service (Amazon S3) buckets. For more information, see Monitoring S3 object scans with Amazon EventBridge.	September 13, 2024
Updated functionality in Runtime Monitoring - Amazon EC2	Runtime Monitoring released a new agent version 1.3.1 for Amazon EC2 resources. For more information about release notes, see GuardDuty security agent for Amazon EC2.	September 12, 2024
Updated functionality in Runtime Monitoring - Amazon ECS-Fargate	Runtime Monitoring released a new agent version 1.3.1 for Amazon ECS-Fargate resources. For more information about release notes, see GuardDuty security agent for AWS Fargate (Amazon ECS only).	September 11, 2024

Updated GuardDuty service-linked role (SLR)	GuardDuty has updated the SLR to include the `ec2:DescribeVpcs` permission in the Amazon EC2 actions. For more information, see Service-linked role permissions for GuardDuty.	August 22, 2024
Significant content addition	GuardDuty added significant content updates to the Malware Protection for S3 feature. • Added new examples of sample notification schema to set up Amazon EventBridge rules to receive notification related to **Malware Protection plan resource status** and **S3 object scan result**. For more information, see Monitoring S3 object scans with Amazon EventBridge. • Added information about Troubleshooting S3 object post-scan tag failures.	August 20, 2024
Updated functionality in GuardDuty Runtime Monitoring - Amazon EC2	Runtime Monitoring released a new agent version 1.3.0 for Amazon EC2 resources. For more information about release notes, see GuardDuty security agent for Amazon EC2.	August 19, 2024

Updated functionality in GuardDuty Runtime Monitoring - Amazon EKS	Runtime Monitoring released a new agent version 1.7.0 for Amazon EKS resources. For more information about release notes, see GuardDuty security agent for Amazon EKS clusters.	August 17, 2024
Significant content addition	GuardDuty added new information about malware detection methodology and scan engines that it uses for the Malware Protection for S3 and Malware Protection for EC2 features. For more information, see GuardDuty malware detection scan engine.	August 15, 2024
New feature - Protecting AI workloads	GuardDuty foundational threat detection and Lambda Protection helps you to better secure and detect threats to AI workloads built on AWS. For more information, see Protecting AI workloads with GuardDuty.	August 14, 2024
Updated functionality in GuardDuty Runtime Monitoring - Fargate (Amazon ECS only)	Runtime Monitoring released a new agent version 1.3.0 for AWS Fargate (Amazon ECS only) resources. For more information about release notes, see GuardDuty security agent for Fargate-ECS.	August 9, 2024

Updated functionality - Malware Protection for S3	GuardDuty Malware Protection for S3 increases the maximum number of S3 buckets quota from 10 to 25 buckets. This quota applies to an AWS account per each AWS Region. For more information, see Malware Protection for S3.	August 8, 2024
Updated - New finding types in Runtime Monitoring	GuardDuty has added two new Runtime Monitoring finding types that will help you detect threats involving suspicious shell creation on the monitored resource, and privilege escalation where a process suspiciously elevates its privileges to root. • Execution:Runtime/SuspiciousShellCreated • PrivilegeEscalation:Runtime/ElevationToRoot	August 6, 2024
Updated - Integrating with AWS Security Hub	AWS Security Hub provides a list of GuardDuty security controls to evaluate your resources, and check your compliance against security industry standards and best practices. For more information, see Using GuardDuty controls in Security Hub.	July 11, 2024

Updated GuardDuty tester script for findings	GuardDuty now supports over 100 findings with different AWS resources in a dedicated account. For more information, see Test GuardDuty findings in dedicated accounts.	June 28, 2024
Updated functionality in Runtime Monitoring	Runtime Monitoring released a new security agent version 1.2.0 for the Amazon EC2 resource. For information about release notes, see GuardDuty security agent for Amazon EC2 instance. For information about updating the security agent to this release version manually, see Managing security agent manually for Amazon EC2 instance.	June 13, 2024

New feature - Malware Protection for S3 Region availability	GuardDuty Malware Protection for S3 is now available in all the commercial Regions where GuardDuty is available. This feature helps you scan newly uploaded objects to Amazon S3 buckets for potential malware and suspicious uploads, and take action to isolate them before they are ingested into downstream processes. For information about enabling Malware Protection for S3, see GuardDuty Malware Protection for S3.	June 12, 2024

New feature - Malware Protection for S3	GuardDuty announces general availability of Malware Protection for S3 that helps you scan newly uploaded objects to Amazon S3 buckets for potential malware and suspicious uploads, and take action to isolate them before they are ingested into downstream processes. This feature is fully managed by AWS. GuardDuty publishes the S3 object scan result to your EventBridge default event bus. You can allow GuardDuty to add tags to your scanned S3 objects. You can build downstream workflows, such as isolation to a quarantine bucket, or define bucket policies using tags that prevent users or applications from accessing certain objects. For more information, see GuardDuty Malware Protection for S3. Presently, it is available in the following Regions: • US East (N. Virginia) • US East (Ohio) • US West (Oregon) • Europe (Ireland) • Europe (Frankfurt) • Europe (Stockholm)	June 11, 2024

	• Asia Pacific (Sydney) • Asia Pacific (Tokyo) • Asia Pacific (Singapore)	
Updated AmazonGuardDutyFullAccess policy	Added permission that allows you to pass an IAM role to GuardDuty when you enable Malware Protection for S3. For more information about this policy update, see AWS managed policy: AmazonGuardDutyFullAccess.	June 10, 2024
Updated functionality in GuardDuty RDS Protection	RDS Protection extends support to monitor the login activity on your RDS for PostgreSQL databases. As part of this expansion, GuardDuty will automatically begin monitoring login data from RDS for PostgreSQL databases for accounts that have already enabled GuardDuty RDS Protection. For more information, see RDS Protection.	June 6, 2024
Updated functionality in GuardDuty Runtime Monitoring - Fargate (Amazon ECS only)	Runtime Monitoring released a new agent version 1.2.0 for AWS Fargate (Amazon ECS only) resources. For more information about release notes, see GuardDuty security agent for Fargate-ECS.	May 31, 2024

Updated functionality in GuardDuty Malware Protection for EC2	For each Amazon EBS volume that is attached to your Amazon EC2 instances and container workloads, GuardDuty Malware Protection for EC2 has increased the size of the EBS volume that it scans to up to 2048 GB. For information about scanning Amazon EBS volumes attached to your instances, see GuardDuty Malware Protection for EC2.	May 29, 2024
Updated functionality in Runtime Monitoring	Runtime Monitoring for Amazon ECS-Fargate resources now supports detecting potential threats on your tasks launched by AWS Batch and AWS CodePipeline. For more information, see How Runtime Monitoring works with Fargate (Amazon ECS only).	May 28, 2024
Updated functionality in Runtime Monitoring	Runtime Monitoring released a new agent version 1.6.1 for Amazon EKS resources. For information about release notes, see EKS add-on agent release history.	May 14, 2024

Expanded Region support for Runtime Monitoring	GuardDuty expands the support for Runtime Monitoring to the Canada West (Calgary) Region. For information about getting started with Runtime Monitoring, see Enabling Runtime Monitoring.	May 7, 2024
Expanded Region support for RDS Protection	GuardDuty expands RDS Protection support to the following AWS Regions: • Canada West (Calgary) • Asia Pacific (Hyderabad) • Europe (Spain) • Europe (Zurich) • Middle East (UAE) • Israel (Tel Aviv) • Asia Pacific (Melbourne) For information about enabling this feature, see RDS Protection.	May 3, 2024
Updated functionality in Runtime Monitoring	Runtime Monitoring released a new agent version 1.1.0 for AWS Fargate (Amazon ECS only) resources. For more information about release notes, see GuardDuty security agent for Fargate-ECS.	May 1, 2024

Updated functionality in Runtime Monitoring	Runtime Monitoring released a new agent version 1.6.0 for Amazon EKS resources. For information about release notes, see EKS add-on agent release history.	April 29, 2024
Support for IPAddressv6	GuardDuty has added IPAddressv6 support for both local and remote IP details. You can use the associate d Filter attributes to filter GuardDuty findings or create suppression rules.	April 18, 2024
Updated console experienc e to configure exporting findings	GuardDuty has updated the console experience to export the findings generated in your AWS accounts, to an Amazon S3 bucket. For more informati on, see Exporting GuardDuty findings.	April 1, 2024
Updated functionality in Runtime Monitoring	Runtime Monitoring released a new security agent version 1.1.0 for the Amazon EC2 resource. This version supports GuardDuty automated agent configura tion in Runtime Monitorin g for Amazon EC2 instances . For information about release notes, see GuardDuty security agent for Amazon EC2 instance.	March 28, 2024

General availability of Runtime Monitoring for Amazon EC2 instances	GuardDuty announces general availability(GA) of Runtime Monitoring for Amazon EC2 instances. Now, you have an option to enable automated agent configuration that permits GuardDuty to install and manage the security agent for your Amazon EC2 instances on your behalf. With GuardDuty automated agent, you can also use inclusion or exclusion tags to inform GuardDuty to install and manage the security agent on selected Amazon EC2 instances only. For more information, see How Runtime Monitoring works with Amazon EC2 instances. **List of new finding types released along with this GA** • Execution:Runtime/SuspiciousTool • Execution:Runtime/SuspiciousCommand • DefenseEvasion:Runtime/SuspiciousCommand • DefenseEvasion:Runtime/PtraceAntiDebugging • Execution:Runtime/MaliciousFileExecuted	March 28, 2024

Amazon GuardDuty has updated the Service-linked role (SLR)	Use AWS Systems Manager actions to manage SSM associations on Amazon EC2 instances when you enable GuardDuty Runtime Monitoring with automated agent for Amazon EC2. When GuardDuty automated agent configuration is disabled, GuardDuty considers only those EC2 instances that have an inclusion tag (`GuardDuty Managed :true`).
	
	• The following list shows the new permissions:
	
	`"ssm:DescribeAssociation",`
	`"ssm:DeleteAssociation",`
	`"ssm:UpdateAssociation",`
	`"ssm:CreateAssociation",`
	`"ssm:StartAssociationsOnce",`
	`"ssm:AddTagsToResource",`
	`"ssm:CreateAssociation",`
	`"ssm:UpdateAssociation",`
	`"ssm:SendCommand",`
	`"ssm:GetCommandInvocation"`
		March 26, 2024

Updated functionality in Runtime Monitoring	With the latest GuardDuty security agent (add-on) v1.5.0 release for Amazon EKS, Runtime Monitoring now supports configuring specific parameters of your GuardDuty security agent, such as CPU and memory settings, `PriorityClass` settings, and DNS policy settings. For more information, see Configuring GuardDuty security agent (EKS add-on) parameters.	March 7, 2024
Updated functionality in Runtime Monitoring	Runtime Monitoring released a new agent version 1.5.0 for Amazon EKS resources. For information about release notes, see EKS add-on agent release history.	March 7, 2024
Support for Canada West (Calgary)	Amazon GuardDuty is now available in the Canada West (Calgary) Region. Some of the protection plans within GuardDuty might not be available in this Region. For the latest information, see Regions and endpoints.	March 6, 2024

Updated functionality in Runtime Monitoring	The GuardDuty security agent versions 1.0.0 and 1.1.0 for Amazon EKS clusters will no longer be supported starting May 14, 2024. For informati on about what steps you can take before the end of standard support, see GuardDuty security agent for Amazon EKS clusters.	February 16, 2024
Updated functionality in Runtime Monitoring	Runtime Monitoring supports the latest Kubernetes version 1.29 with the existing security agent version 1.4.1. The support has been available since the launch of this Kubernete s version. For information about supported Kubernete s versions, see Kubernete s versions supported by GuardDuty security agent.	February 16, 2024

Updated functionality in Runtime Monitoring - Regional availability	GuardDuty Runtime Monitoring now supports shared Amazon VPC within the same AWS Organizations. GuardDuty service-linked role (SLR) has a new permission – `organizations:DescribeOrganization` that helps retrieving the organization ID for the shared Amazon VPC account to set the endpoint policy. For information about prerequisites to using a shared Amazon VPC endpoint in Runtime Monitoring, see Support for shared Amazon VPC. This capability is available in all the Regions where GuardDuty supports Runtime Monitoring.	February 12, 2024

Updated functionality in Runtime Monitoring - Regional availability	GuardDuty Runtime Monitoring now supports shared Amazon VPC within the same AWS Organizations. GuardDuty service-linked role (SLR) has a new permission – `organizations:DescribeOrganization` that helps retrieving the organization ID for the shared Amazon VPC account to set the endpoint policy. For information about prerequisites to using a shared Amazon VPC endpoint in Runtime Monitoring, see Support for shared Amazon VPC. Presently, this capability is available in some of the AWS Regions. For more information, see Regions and endpoints.	February 9, 2024
Updated functionality with support for new AWS Regions – Malware Protection for EC2	Malware Protection for EC2 now supports scanning the EBS volumes encrypted with AWS managed keys in the US West (Oregon) Region.	February 6, 2024

Updated functionality with support for new AWS Regions – Malware Protection for EC2	Malware Protection for EC2 now supports scanning the EBS volumes encrypted with AWS managed keys in the following AWS Regions: • Asia Pacific (Singapore) (`ap-southeast-1`) • Europe (Frankfurt) (`eu-central-1`) • Asia Pacific (Osaka) (`ap-northeast-3`) • US East (Ohio) (`us-east-2`) • Europe (Milan) (`eu-south-1`) • Asia Pacific (Tokyo) (`ap-northeast-1`) • Asia Pacific (Seoul) (`ap-northeast-2`) • Canada (Central) (`ca-central-1`) • Europe (Ireland) (`eu-west-1`) • US East (N. Virginia) (`us-east-1`)	February 5, 2024

Updated functionality in Runtime Monitoring	GuardDuty Runtime Monitoring has released a new GuardDuty security agent version (v1.0.2) for Amazon EC2 instances. This agent version includes support for the latest Amazon ECS AMIs. For more information about agent release history, see GuardDuty security agent for Amazon EC2 instances.	February 2, 2024

Updated functionality with support for new AWS Regions – Malware Protection for EC2	Malware Protection for EC2 now supports scanning the Amazon EBS volumes encrypted with AWS managed keys in the following AWS Regions: • Europe (London) (`eu-west-2`) • Europe (Stockholm) (`eu-north-1`) • Asia Pacific (Hong Kong) (`ap-east-1`) • Africa (Cape Town) (`af-south-1`) • Middle East (Bahrain) (`me-south-1`) • Asia Pacific (Hyderabad) (`ap-south-2`) • Europe (Spain) (`eu-south-2`) • Asia Pacific (Melbourne) (`ap-southeast-4`) • Asia Pacific (Sydney) (`ap-southeast-2`) • Israel (Tel Aviv) (`il-central-1`)	January 31, 2024

Updated Managing accounts with AWS Organizations	Reorganized the content under Managing accounts with AWS Organizations., added steps to change the delegated GuardDuty administrator account, and updated Understanding the relationship between GuardDuty administrator account and member accounts.	January 30, 2024
Updated functionality with support for new AWS Regions	Malware Protection for EC2 now supports scanning the EBS volumes encrypted with AWS managed keys in the following AWS Regions: • Asia Pacific (Jakarta) (`ap-southeast-3`) • US West (N. California) (`us-west-1`) • Middle East (UAE) (`me-central-1`) • Europe (Zurich) (`eu-central-2`) • Asia Pacific (Mumbai) (`ap-south-1`) • South America (São Paulo) (`sa-east-1`)	January 29, 2024

Updated functionality in Malware Protection for EC2	Malware Protection for EC2 now supports scanning the EBS volumes encrypted using AWS managed keys. Malware Protection for EC2 service-linked role (SLR) has two new permissions – `GetSnapshotBlock` and `ListSnapshotBlocks`. These permissions will help GuardDuty fetch the snapshot of an EBS volume (encrypted using AWS managed key) from your AWS account and copy it to the GuardDuty service account before starting the malware scan. Presently, this functionality is available in Europe (Paris) (`eu-west-3`) only. For more information, see Supported volumes for malware scan.	January 25, 2024
Updated functionality in Runtime Monitoring	GuardDuty Runtime Monitoring has released a new GuardDuty security agent version (v1.0.1) with general performance tuning and enhancements. For more information about agent release history, see GuardDuty security agent for Amazon EC2 instances.	January 23, 2024

Updated functionality in Runtime Monitoring	Runtime Monitoring released a new agent version 1.4.1 for Amazon EKS resources. For more information, see EKS add-on agent release history.	January 16, 2024
Runtime Monitoring released new agent v1.4.0 for Amazon EKS resources	Runtime Monitoring released a new agent version 1.4.0 for Amazon EKS resources. For more information, see EKS add-on agent release history.	December 21, 2023

Added S3 and AWS CloudTrail machine learning (ML)-based findings types to the Europe (Zurich) , Europe (Spain), Asia Pacific (Hyderabad), Asia Pacific (Melbourne), and Israel (Tel Aviv)	The following S3 and CloudTrail findings that identify the anomalous behavior using the GuardDuty 's anomaly detection machine learning (ML) model are now available in the Europe (Zurich) , Europe (Spain), Asia Pacific (Hyderabad), Asia Pacific (Melbourne), and Israel (Tel Aviv) Regions: • Discovery:S3/AnomalousBehavior • Impact:S3/AnomalousBehavior.Write • Impact:S3/AnomalousBehavior.Delete • Impact:S3/AnomalousBehavior.Permission • Exfiltration:S3/AnomalousBehavior • Exfiltration:IAMUser/AnomalousBehavior • Impact:IAMUser/AnomalousBehavior • CredentialAccess:IAMUser/AnomalousBehavior • DefenseEvasion:IAMUser/AnomalousBehavior • InitialAccess:IAMUser/AnomalousBehavior • Persistence:IAMUser/AnomalousBehavior	December 21, 2023

	• PrivilegeEscalation:IAMUser/AnomalousBehavior • Discovery:IAMUser/AnomalousBehavior	
GuardDuty supports 50,000 member accounts through AWS Organizations	A delegated GuardDuty administrator can now manage a maximum of 50,000 member accounts through AWS Organizations. This also includes a maximum of 5000 member accounts that associated with the GuardDuty administrator account by invitation.	December 20, 2023
GuardDuty Runtime Monitoring support expanded to 19 AWS Regions	Runtime Monitoring is now available in Asia Pacific (Jakarta), Europe (Paris), Asia Pacific (Osaka), Asia Pacific (Seoul), Middle East (Bahrain), Europe (Spain), Asia Pacific (Hyderabad), Asia Pacific (Melbourne), Israel (Tel Aviv), US West (N. California), Europe (London), Asia Pacific (Hong Kong), Europe (Milan), Middle East (UAE), South America (São Paulo), Asia Pacific (Mumbai), Canada (Central), Africa (Cape Town), Europe (Zurich).	December 6, 2023

GuardDuty expands Runtime Monitoring capability	In addition to detecting threats to your Amazon EKS clusters, GuardDuty announces general availability of Runtime Monitoring to detect threats to your Amazon ECS workloads and a preview release to detect threats to your Amazon EC2 instances. For more information about which AWS Regions presently support Runtime Monitoring, see Regions and endpoints.	November 26, 2023

Amazon GuardDuty has updated the Service-linked role (SLR)	GuardDuty has added new permissions to use Amazon ECS actions to manage and retrieve information about the Amazon ECS clusters, and manage the Amazon ECS account setting with `guarddutyActivate` . The actions pertaining to Amazon ECS also retrieve the information about the tags associated with GuardDuty.
	
	• The following permissions have been added as a part of GuardDuty expanding the Runtime Monitoring capability:
	
	`"ecs:ListClusters",`
	`"ecs:DescribeClusters",`
	`"ecs:PutAccountSettingDefault"`
		November 26, 2023

Updated the AWS managed policies	GuardDuty added a new permission, `organizations:ListAccounts` to the AWS managed policy: AmazonGuardDutyFullAccess and AmazonGuardDutyReadOnlyAccess.	November 16, 2023

[GuardDuty released new finding types that use EKS Audit Log Monitoring.](#)	EKS Audit Log Monitoring now supports the following finding types in Asia Pacific (Melbourne)(`ap-southeast-4`). • CredentialAccess:Kubernetes/AnomalousBehavior.SecretsAccessed • PrivilegeEscalation:Kubernetes/AnomalousBehavior.RoleBindingCreated • Execution:Kubernetes/AnomalousBehavior.ExecInPod • PrivilegeEscalation:Kubernetes/AnomalousBehavior.WorkloadDeployed!PrivilegedContainer • PrivilegeEscalation:Kubernetes/AnomalousBehavior.WorkloadDeployed!ContainerWithSensitiveMount • Execution:Kubernetes/AnomalousBehavior.WorkloadDeployed • PrivilegeEscalation:Kubernetes/AnomalousBehavior.RoleCreated • Discovery:Kubernetes/AnomalousBehavior.PermissionChecked	November 11, 2023

[GuardDuty released new finding types that use EKS Audit Log Monitoring.](#)	EKS Audit Log Monitoring now supports the following finding types in Asia Pacific (Hyderabad) (`ap-south-2`), Europe (Zurich) (`eu-central-2`), and Europe (Spain) (`eu-south-2`) Regions. • CredentialAccess:Kubernetes/AnomalousBehavior.SecretsAccessed • PrivilegeEscalation:Kubernetes/AnomalousBehavior.RoleBindingCreated • Execution:Kubernetes/AnomalousBehavior.ExecInPod • PrivilegeEscalation:Kubernetes/AnomalousBehavior.WorkloadDeployed!PrivilegedContainer • PrivilegeEscalation:Kubernetes/AnomalousBehavior.WorkloadDeployed!ContainerWithSensitiveMount • Execution:Kubernetes/AnomalousBehavior.WorkloadDeployed • PrivilegeEscalation:Kubernetes/AnomalousBehavior.RoleCreated • Discovery:Kubernetes/AnomalousBehavior.PermissionChecked	November 10, 2023

GuardDuty released new finding types that use EKS Audit Log Monitoring.	EKS Audit Log Monitoring now supports the following finding types. These finding types are not yet available in Asia Pacific (Hyderabad) (`ap-south-2`), Europe (Zurich) (`eu-central-2`), Europe (Spain) (`eu-south-2`), and Asia Pacific (Melbourne) (`ap-southeast-4`) Regions. • CredentialAccess:Kubernetes/AnomalousBehavior.SecretsAccessed • PrivilegeEscalation:Kubernetes/AnomalousBehavior.RoleBindingCreated • Execution:Kubernetes/AnomalousBehavior.ExecInPod • PrivilegeEscalation:Kubernetes/AnomalousBehavior.WorkloadDeployed!PrivilegedContainer • PrivilegeEscalation:Kubernetes/AnomalousBehavior.WorkloadDeployed!ContainerWithSensitiveMount • Execution:Kubernetes/AnomalousBehavior.WorkloadDeployed • PrivilegeEscalation:Kubernetes/AnomalousBehavior.RoleCreated	November 8, 2023

	• Discovery:Kubernetes/AnomalousBehavior.PermissionChecked	
EKS Runtime Monitoring released new agent v1.3.1	EKS Runtime Monitoring released a new agent version 1.3.1 that includes important security patches and updates.	October 23, 2023
New filter attribute for finding	GuardDuty has added a new criteria to filter the generated findings. DNS request domain suffix provides the second- and top-level domain involved in the activity that prompted GuardDuty to generate the finding.	October 17, 2023
EKS Runtime Monitoring released new agent v1.3.0 that supports Kubernetes version 1.28	EKS Runtime Monitoring released a new agent version 1.3.0 that supports Kubernetes version 1.28. Added support for Ubuntu. For more information, see EKS add-on agent release history.	October 5, 2023

Added S3 and AWS CloudTrail machine learning (ML)-based findings types to the Asia Pacific (Jakarta) and Middle East (UAE) Regions	The following S3 and CloudTrail findings that identify the anomalous behavior using the GuardDuty 's anomaly detection machine learning (ML) model are now available in the Asia Pacific (Jakarta) and Middle East (UAE) Regions: • Discovery:S3/AnomalousBehavior • Impact:S3/AnomalousBehavior.Write • Impact:S3/AnomalousBehavior.Delete • Impact:S3/AnomalousBehavior.Permission • Exfiltration:S3/AnomalousBehavior • Exfiltration:IAMUser/AnomalousBehavior • Impact:IAMUser/AnomalousBehavior • CredentialAccess:IAMUser/AnomalousBehavior • DefenseEvasion:IAMUser/AnomalousBehavior • InitialAccess:IAMUser/AnomalousBehavior • Persistence:IAMUser/AnomalousBehavior • PrivilegeEscalation:IAMUser/AnomalousBehavior	September 20, 2023

	• Discovery:IAMUser/AnomalousBehavior	
GuardDuty EKS Runtime Monitoring introduces managing GuardDuty security agent at the cluster level	EKS Runtime Monitoring adds support to manage the GuardDuty security agent for individual EKS clusters to monitor the runtime events from only these selective clusters. EKS Runtime Monitoring extends this capability with the support of tags.	September 13, 2023
GuardDuty Malware Protection for EC2 extends support to more AWS Regions	Malware Protection for EC2 is now available in Asia Pacific (Hyderabad), Asia Pacific (Melbourne), Europe (Zurich), and Europe (Spain).	September 11, 2023

GuardDuty is now available in Israel (Tel Aviv) Region	Added Israel (Tel Aviv) Region to the list of AWS Regions where GuardDuty is now available. The following protection plans are also available in the Israel (Tel Aviv) Region: • EKS Protection includes both EKS Audit Log Monitoring and EKS Runtime Monitoring. • Lambda Protection. • Malware Protection for EC2. • S3 Protection. For more information about protection plan availability in the Israel (Tel Aviv) Region, see Regions and endpoints.	August 24, 2023
GuardDuty added auto-enable configuration for your organization at protection plan level	Update organization configuration for the protection plans in your Region. Possible configuration options are either enable for all accounts, auto-enable for new accounts, or do not auto-enable for any account in your organization.	August 16, 2023

S3 finding types which identify anomalous behavior using GuardDuty's anomaly detection machine learning (ML) model are now available in Asia Pacific (Osaka)	The following findings types are now available in the Asia Pacific (Osaka) Region: • Discovery:S3/AnomalousBehavior • Impact:S3/AnomalousBehavior.Write • Impact:S3/AnomalousBehavior.Delete • Impact:S3/AnomalousBehavior.Permission • Exfiltration:S3/AnomalousBehavior	August 10, 2023
EKS Runtime Monitoring is now available in Asia Pacific (Melbourne)	EKS Runtime Monitoring within GuardDuty EKS Protection provides runtime threat detection for your Amazon EKS clusters in AWS environment. It is now supported in the Asia Pacific (Melbourne) Region.	August 8, 2023
Updated the list of GuardDuty findings that invoke GuardDuty-initiated malware scan	Certain EKS Runtime Monitoring finding types can now invoke GuardDuty-initiated malware scan in your AWS account.	July 19, 2023

GuardDuty supports 10,000 member accounts through AWS Organizations	A GuardDuty administrator account can now manage a maximum of 10,000 member accounts through AWS Organizations. This also includes a maximum of 5000 member accounts that associated with the GuardDuty administrator account by invitation.	June 29, 2023
EKS Runtime Monitoring announces three new finding types.	EKS Runtime Monitoring supports three new finding types that are based on the process injection technique . The new finding types are DefenseEvasion:Runtime/ProcessInjection.Proc, DefenseEvasion:Runtime/ProcessInjection.Ptrace, and DefenseEvasion:Runtime/ProcessInjection.VirtualMemoryWrite.	June 22, 2023
EKS Runtime Monitoring released new agent v1.2.0 that supports Kubernetes version 1.27	EKS Runtime Monitoring released a new agent version 1.2.0 that also supports ARM64-based instances. Added support for Bottlerocket. For more information, see EKS add-on agent release history.	June 16, 2023

GuardDuty console provides a summarized view of your findings.	The summary dashboard in the GuardDuty console provides an aggregated view of the GuardDuty findings. Presently, the dashboard displays data through various widgets for the last 10,000 findings generated for your account (or member accounts if you're a GuardDuty administrator account) for the current Region.	June 12, 2023
EKS Audit Log Monitoring is now available in Asia Pacific (Hyderabad), Asia Pacific (Melbourne), Europe (Zurich), and Europe (Spain)	Enable EKS Audit Log Monitoring (in EKS Protection) for your accounts to monitor EKS audit logs from your Amazon EKS clusters and analyze them for potentially malicious and suspicious activity.	June 1, 2023
EKS Audit Log Monitoring is now available in Middle East (UAE)	EKS Audit Log Monitoring is now available in Middle East (UAE). Enable EKS Audit Log Monitoring for your accounts to monitor EKS audit logs from your Amazon EKS clusters and analyze them for potentially malicious and suspicious activity.	May 3, 2023

GuardDuty Malware Protection for EC2 announces On-demand malware scan	Malware Protection for EC2 helps you detect the potential presence of malware in the Amazon EBS volumes attached to your Amazon EC2 instances and container workloads. It now offers two types of scans – GuardDuty initiated and on-demand. GuardDuty-initiated malware scan initiates an agentless scan in the Amazon EBS volumes automatically only when GuardDuty generates one of the Findings that invoke GuardDuty-initiated malware scan. You can initiate an On-demand malware scan for Amazon EC2 instances in your account by providing the Amazon Resource Name (ARN) associated to that Amazon EC2 instance. For more information about how both the scan types differ, see Malware Protection for EC2. • GuardDuty-initiated malware scan • On-demand malware scan	April 27, 2023

GuardDuty announces Lambda Protection	Lambda Protection helps you identify potential security threats in your AWS Lambda functions. • Lambda Protection finding types • Remediating a potentially compromised Lambda function	April 20, 2023
GuardDuty is now available in the Asia Pacific (Melbourne) Region	Added Asia Pacific (Melbourne) to the list of AWS Regions where GuardDuty is available. For information about which features are available in this Region, see Regions and endpoints.	April 19, 2023
GuardDuty added 3 new EC2 findings types	GuardDuty introduces new finding types to detect the use of external DNS resolvers and encrypted DNS technologies. For information about AWS Regions where these finding types are supported, see Regions and endpoints. • DefenseEvasion:EC2/UnusualDNSResolver • DefenseEvasion:EC2/UnusualDoHActivity • DefenseEvasion:EC2/UnusualDoTActivity	April 5, 2023

GuardDuty announces EKS Runtime Monitoring in EKS Protection	EKS Runtime Monitoring within EKS Protection provides runtime threat detection for your Amazon EKS clusters in AWS environment. It uses an Amazon EKS add-on agent (`aws-guardduty-agent`) that collects Runtime events from your EKS workloads. After GuardDuty receives these runtime events, it monitors and analyzes them to identify potential suspicious security threats. For more information, see Finding details and EKS Runtime Monitoring finding types.	March 30, 2023

GuardDuty adds a new functionality – `autoEnableOrganizationMembers`	Amazon GuardDuty adds a new organization configuration option that helps GuardDuty administrator accounts audit and enforce (if required) that GuardDuty is enabled for `ALL` the members of their organization. The best practice now is to use `autoEnableOrganizationMembers` instead of `autoEnable`. `autoEnable` is deprecated but still supported. The following APIs are impacted by this new functionality: • DescribeOrganizationConfiguration • UpdateOrganizationConfiguration • DisassociateMembers • DeleteMembers • DisassociateFromAdministratorAccount • StopMonitoringMembers	March 23, 2023

The RDS Protection feature in Amazon GuardDuty is now generally available	GuardDuty RDS Protection monitors and profiles RDS login activity to identify suspicious login behavior on your Amazon Aurora database instances. For information about which AWS Regions support RDS Protection, see Regions and endpoints.	March 16, 2023
GuardDuty announces feature activation	Historically, the GuardDuty API allowed configuration of both features and data sources, but now, all new GuardDuty protection types will be configured as features and not as data sources. GuardDuty still supports the data sources via API but will not add a new API. Features activation affects the behavior of the APIs used to enable GuardDuty or a protection type within GuardDuty. If you manage your GuardDuty accounts through API, SDK, or CFN template, see GuardDuty API changes in March 2023.	March 16, 2023
GuardDuty Malware Protection for EC2 is now available in Middle East (UAE) Region	The Malware Protection for EC2 feature in GuardDuty is supported in the Middle East (UAE) Region. For more information, see Regions and endpoints.	March 13, 2023

Amazon GuardDuty has updated the Service-linked role (SLR)	GuardDuty added the following new permissions to support the upcoming GuardDuty EKS Runtime Monitoring feature. • Use Amazon EKS actions to manage and retrieve information about the EKS clusters, and manage EKS add-ons on EKS clusters. The EKS actions also retrieve the information about the tags associated with GuardDuty. `"eks:ListClusters", "eks:DescribeCluster", "ec2:DescribeVpcEndpointServices", "ec2:DescribeSecurityGroups"`	March 8, 2023
Amazon GuardDuty has updated the Service-linked role (SLR)	The GuardDuty SLR has been updated to allow creation of Malware Protection for EC2 SLR after Malware Protection for EC2 has been enabled.	February 21, 2023
GuardDuty requires TLS v1.2 or later	To communicate with AWS resources, GuardDuty requires and supports TLS v1.2 or later. For more information, see Data protection and Infrastructure security.	February 14, 2023

GuardDuty is now available in Asia Pacific (Hyderabad) Region	Added Asia Pacific (Hyderabad) Region to the list of AWS Regions where GuardDuty is available. For more information, see Regions and endpoints.	February 14, 2023
Amazon GuardDuty User Guide is aligned with IAM best practices	Updated guide to align with the IAM best practices. For more information, see Security best practices in IAM.	February 10, 2023
GuardDuty is now available in Europe (Spain) Region	Added Europe (Spain) to the list of AWS Regions where GuardDuty is available. For more information, see Regions and endpoints.	February 8, 2023
GuardDuty is now available in Europe (Zurich) Region	Added Europe (Zurich) to the list of AWS Regions where GuardDuty is available. For more information, see Regions and endpoints.	December 12, 2022
Preview release of a new feature – GuardDuty RDS Protection	GuardDuty RDS Protection monitors and profiles RDS login activity to identify suspicious login behavior on your Amazon Aurora database instances. Presently, it is available for a preview release in five AWS Regions. For more information, see Regions and endpoints.	November 30, 2022

GuardDuty is now available in Middle East (UAE) Region	Added Middle East (UAE) to the list of AWS Regions where GuardDuty is available . For more information, see Regions and endpoints.	October 6, 2022

Added content for a new feature – GuardDuty Malware Protection for EC2	GuardDuty Malware Protectio n for EC2 is an optional enhancement to Amazon GuardDuty. While GuardDuty identifies the resources at risk, Malware Protection for EC2 detects the malware that may be the source of the compromise. With Malware Protection for EC2 enabled, whenever GuardDuty detects suspicious behavior on an Amazon EC2 instance or a container workload indicativ e of malware, GuardDuty Malware Protection for EC2 initiates an agentless scan on the EBS volumes attached to impacted EC2 instance or container workloads to detect the presence of malware. For information about how Malware Protection for EC2 works and configuring this feature, see GuardDuty Malware Protection for EC2. • For information about Malware Protection for EC2 findings, see Finding details. • For information about remediating the compromis ed EC2 instance and a standalone container, see	July 26, 2022

	Remediating security issues discovered by GuardDuty. • For information about auditing CloudWatch logs for malware scans and reasons for skipping a resource during malware scan, see Understanding CloudWatch Logs and skip reasons. • For information about false positive threat detection s, see Reporting false positives in GuardDuty Malware Protection for EC2.	
Retired one finding type	Exfiltration:S3/ObjectRead.Unusual has been retired.	July 5, 2022

Added new S3 finding types which identify anomalous behavior using GuardDuty's anomaly detection machine learning (ML) model.	Added the following new S3 finding types. These finding types identify if an API request invoked an IAM entity in an anomalous way. The ML model evaluates all API requests in your account and identifies anomalous events that are associate d with techniques used by adversaries. To learn more about each of these new findings, see S3 finding types. • Discovery:S3/Anoma lousBehavior • Impact:S3/Anomalou sBehavior.Write • Impact:S3/Anomalou sBehavior.Delete • Impact:S3/Anomalou sBehavior.Permission • Exfiltration:S3/An omalousBehavior	July 5, 2022

Added GuardDuty EKS Protection content for GuardDuty	GuardDuty can now generate findings for your Amazon EKS resources through the monitoring of EKS audit logs. To learn how to configure this feature, see EKS Protection in Amazon GuardDuty. For a list of findings GuardDuty can generate for Amazon EKS resources, see Kubernetes findings. New remediation guidance has been added to support remediating these findings in the Kubernetes finding remediation guide.	January 25, 2022
Added 1 new finding	A new finding UnauthorizedAccess:IAMUser/InstanceCredentialExfiltration.InsideAWS has been added. This finding informs you when your instance credentials are accessed by an AWS account outside your AWS environment.	January 20, 2022
Updated the finding types to help identify issues related to log4j	Amazon GuardDuty has updated the following finding types to help identify and prioritize issues related to CVE-2021-44228 and CVE-2021-45046: Backdoor:EC2/C&CActivity.B; Backdoor:EC2/C&CActivity.B!DNS; Behavior:EC2/NetworkPortUnusual.	December 22, 2021

Finding Changes	UnauthorizedAccess:IAMUser/InstanceCredentialExfiltration has been changed to UnauthorizedAccess:IAMUser/InstanceCredentialExfiltration.OutsideAWS. This improved version of the finding learns the typical locations your credentials are used from to reduce findings from traffic routed through on premise networks. UnauthorizedAccess:IAMUser/InstanceCredentialExfiltration.OutsideAWS	September 7, 2021
Update to GuardDuty SLR	The GuardDuty SLR has been updated with new actions to improve finding accuracy.	August 3, 2021
Added data source information for each finding type.	Finding descriptions now contain information about data sources that GuardDuty uses to generate that finding.	May 10, 2021

Retired 13 finding types.	13 findings have been retired to be replaced with new AnomalousBehavoir findings. Persistence:IAMUser/NetworkPermissions, Persistence:IAMUser/ResourcePermissions, Persistence:IAMUser/UserPermissions, PrivilegeEscalation:IAMUser/AdministrativePermissions, Recon:IAMUser/NetworkPermissions, Recon:IAMUser/ResourcePermissions, Recon:IAMUser/UserPermissions, ResourceConsumption:IAMUser/ComputeResources, Stealth:IAMUser/LoggingConfigurationModified, Discovery:S3/BucketEnumeration.Unusual, Impact:S3/ObjectDelete.Unusual, Impact:S3/PermissionsModification.Unusual, and UnauthorizedAccess:IAMUser/ConsoleLogin.	March 12, 2021

Added 8 new finding types for anomalous behavior.	Added 8 new IAMUser finding types based on anomalous behavior for IAM principals. CredentialAccess:IAMUser/AnomalousBehavior, DefenseEvasion:IAMUser/AnomalousBehavior, Discovery:IAMUser/AnomalousBehavior, Exfiltration:IAMUser/AnomalousBehavior, Impact:IAMUser/AnomalousBehavior, InitialAccess:IAMUser/AnomalousBehavior, Persistence:IAMUser/AnomalousBehavior, PrivilegeEscalation:IAMUser/AnomalousBehavior.	March 12, 2021
Added EC2 findings based on domain reputation.	Added 4 new Impact finding types based on domain reputation. Impact:EC2/AbusedDomainRequest.Reputation , Impact:EC2/BitcoinDomainRequest.Reputation, Impact:EC2/MaliciousDomainRequest.Reputation. Also added a new EC2 finding for C&CActivity. Impact:EC2/SuspiciousDomainRequest.Reputation	January 27, 2021

Added 4 new finding types.	Added 3 new S3 Malicious IPCaller findings. Discovery :S3/MaliciousIPCaller, Exfiltrat ion:S3/MaliciousIPCaller, Impact:S3/MaliciousIPCaller. Also added a new EC2 finding for C&CActivity. Backdoor: EC2/C&CActivity.B	December 21, 2020
Retired the Unauthori zedAccess:EC2/TorIPCaller finding type.	The UnauthorizedAccess:EC2/ TorIPCaller finding type is now retired from GuardDuty. Learn more.	October 1, 2020
Added the Impact:EC2/ WinRmBruteForce finding type.	Added a new Impact finding, Impact:EC2/WinRmBr uteForce. Learn more.	September 17, 2020
Added the Impact:EC2/ PortSweep finding type.	Added a new Impact finding, Impact:EC2/PortSweep. Learn more.	September 17, 2020
GuardDuty is now available in the Africa (Cape Town) and Europe (Milan) Regions.	Added Africa (Cape Town) and Europe (Milan) to the list of AWS Regions in which GuardDuty is available. Learn more	July 31, 2020

Added new usage details for monitoring GuardDuty costs.	You can now use new metrics to query GuardDuty usage cost data for your account and accounts you manage. A new overview of usage costs is available in the console at https://console.aws.amazon.com/guardduty/. More detailed information can be accessed through the API.	July 31, 2020
Added content covering S3 protection through S3 data event monitoring in GuardDuty.	GuardDuty S3 Protection is now available through the monitoring of S3 data plane events as a new data source. New accounts will have this feature enabled automatically. If you are already using GuardDuty you can enable the new data source for yourself or your member accounts.	July 31, 2020
Added 14 new S3 Findings.	14 new S3 finding types have been added for S3 control plane and data plane sources.	July 31, 2020

Added additional support for S3 findings and changed 2 existing finding types names.	GuardDuty findings now include more details for findings involving S3 buckets. Existing finding types that were related to S3 activity have been renamed: Policy:IAMUser/S3BlockPublicAccessDisabled has been changed to Policy:S3/BucketBlockPublicAccessDisabled. Stealth:IAMUser/S3ServerAccessLoggingDisabled has been changed to Stealth:S3/ServerAccessLoggingDisabled.	May 28, 2020
Added content for AWS Organizations integration.	GuardDuty now integrates with AWS Organizations delegated administrators to allow you to manage GuardDuty accounts within your organization. When you set a delegated administrator as your GuardDuty administrator account you can automatically enable GuardDuty for any organization member to be managed by the delegated administrator account. You can also automatically enable GuardDuty in new AWS Organizations member accounts. Learn more.	April 20, 2020

Added content for the export findings feature.	Added content that describes the **Export Findings** feature of GuardDuty.	November 14, 2019
Added the UnauthorizedAccess:EC2/MetadataDNSRebind finding type.	Added a new Unauthorized finding, UnauthorizedAccess:EC2/MetadataDNSRebind. Learn more.	October 10, 2019
Added the Stealth:IAMUser/S3ServerAccessLoggingDisabled finding type.	Added a new Stealth finding, Stealth:IAMUser/S3ServerAccessLoggingDisabled. Learn more.	October 10, 2019
Added the Policy:IAMUser/S3BlockPublicAccessDisabled finding type.	Added a new Policy finding, Policy:IAMUser/S3BlockPublicAccessDisabled. Learn more.	October 10, 2019
Retired the Backdoor:EC2/XORDDOS finding type.	The Backdoor:EC2/XORDDOS finding type is now retired from GuardDuty.Learn more	June 12, 2019
Added the PrivilegeEscalation finding type.	The PrivilegeEscalation finding type detects when users attempt to assign escalated, more permissive privileges to their accounts. Learn more	May 14, 2019
GuardDuty is now available in the Europe (Stockholm) Region.	Added Europe (Stockholm) to the list of AWS Regions in which GuardDuty is available. Learn more	May 9, 2019

Added a new finding type, Recon:EC2/PortProbeEMRUnprotectedPort.	This finding informs you that an EMR-related sensitive port on an EC2 Instance is not blocked and is being actively probed. Learn more	May 8, 2019
Added 5 new finding types that detect if your EC2 instances are potentially being used for denial of service (DoS) attacks.	These findings inform you of EC2 instances in your environment that are behaving in a manner that may indicate they is being used to perform Denial of Service (DoS) attacks. Learn more	March 8, 2019
Added a new finding type: Policy:IAMUser/RootCredentialUsage	Policy:IAMUser/RootCredentialUsage finding type informs you that the root user sign-in credentials of your AWS account are being used to make programmatic requests to AWS services. Learn more	January 24, 2019
UnauthorizedAccess:IAMUser/UnusualASNCaller finding type has been retired	The UnauthorizedAccess:IAMUser/UnusualASNCaller finding type has been retired. You will now be notified about activity invoked from unusual networks via other active GuardDuty finding types. The generated finding type will be based on the category of the API that was invoked from an unusual network. Learn more	December 21, 2018

Added two new finding types: PenTest:IAMUser/ParrotLinux and PenTest:IAMUser/PentooLinux	PenTest:IAMUser/ParrotLinux finding type informs you that a computer running Parrot Security Linux is making API calls using credentials that belong to your AWS account. PenTest:IAMUser/PentooLinux finding type informs you that a machine running Pentoo Linux is making API calls using credentials that belong to your AWS account. Learn more	December 21, 2018
Added support for the Amazon GuardDuty announcements SNS topic	You can now subscribe to the GuardDuty announcements SNS topic to receive notifications about newly released finding types, updates to the existing finding types, and other functionality changes. Notifications are available in all formats that Amazon SNS supports. Learn more	November 21, 2018

Added two new finding types: UnauthorizedAccess:EC2/TorClient and UnauthorizedAccess:EC2/TorRelay	UnauthorizedAccess:EC2/TorClient finding type informs you that an EC2 instance in your AWS environment is making connections to a Tor Guard or an Authority node. UnauthorizedAccess:EC2/TorRelay finding type informs you that an EC2 instance in your AWS environment is making connections to a Tor network in a manner that suggests that it's acting as a Tor relay. Learn more	November 16, 2018
Added a new finding type: CryptoCurrency:EC2/BitcoinTool.B	This finding informs you that an EC2 instance in your AWS environment is querying a domain name that is associated with Bitcoin, or other cryptocurrency-related activity. Learn more	November 9, 2018
Added support for updating the frequency of notifications sent to CloudWatch Events	You can now update the frequency of notifications sent to CloudWatch Events for the subsequent occurrences of existing findings. Possible values are 15 minutes, 1 hour, or the default 6 hours. Learn more	October 9, 2018
Added Region support	Added Region support for AWS GovCloud (US-West) Learn more	July 25, 2018

Added support for AWS CloudFormation StackSets in GuardDuty	You can use the Enable Amazon GuardDuty template to enable GuardDuty simultaneously in multiple accounts. Learn more	June 25, 2018
Added support for GuardDuty auto-archive rules	Customers can now build granular auto-archive rules for suppression of findings. For findings that match an auto-archive rule, GuardDuty automatically marks them as archived. This enables customers to further tune GuardDuty to keep only relevant findings in the current findings table. Learn more	May 4, 2018
GuardDuty is available in the Europe (Paris) Region	GuardDuty is now available in Europe (Paris), allowing you to extend continuou s security monitoring and threat detection in this Region. Learn more	March 29, 2018
Creating GuardDuty administr ator account and member accounts through AWS CloudFormation is now supported.	For more information, see `AWS::GuardDuty::master` and `AWS::GuardDuty::member`.	March 6, 2018
Added nine new CloudTrail-based anomaly detections.	These new finding types are automatically enabled in GuardDuty in all supported Regions. Learn more	February 28, 2018

Added three new threat intelligence detections (finding types).	These new finding types are automatically enabled in GuardDuty in all supported Regions. Learn more	February 5, 2018
Limit increase for GuardDuty member accounts.	With this release, you can have up to 1000 GuardDuty member accounts added per AWS account (GuardDut y administrator account account). Learn more	January 25, 2018
Changes in upload and further management of trusted IP lists and threat lists for GuardDuty administr ator account and member accounts.	With this release, Users from administrator account GuardDuty accounts can upload and manage trusted IP lists and threat lists. Users from member GuardDuty accounts can't upload and manage lists. Trusted IP lists and threat lists that are uploaded by the administr ator account account are imposed on GuardDuty functionality in its member accounts. Learn more	January 25, 2018

Earlier updates

Change	Description	Date
Initial publication	Initial publication of the Amazon GuardDuty User Guide.	November 28, 2017