Amazon GuardDuty

Amazon GuardDuty User Guide
# Table of Contents

What is GuardDuty .......................................................... 1
  Pricing for GuardDuty .................................................. 1
  Accessing GuardDuty .................................................. 1

Data Sources .................................................................. 2
  AWS CloudTrail event logs .......................................... 2
  VPC Flow Logs .......................................................... 3
  DNS logs .................................................................. 3

Concepts and Terminology .......................................... 4

Security .......................................................................... 6
  Data Protection .......................................................... 6
    Encryption at Rest ...................................................... 7
    Encryption in Transit ................................................ 7
  Managing Access ....................................................... 7
    Permissions Required to Enable GuardDuty ................. 7
    Using a Service-Linked Role to Delegate Permissions to GuardDuty ................................................ 8
    Using IAM Policies to Delegate Access to GuardDuty to IAM Identities ......................................... 9
  Logging with CloudTrail ............................................ 15
    GuardDuty Information in CloudTrail ....................... 15
    Example: GuardDuty Log File Entries ....................... 15
  GuardDuty Service-Linked Role .................................. 16
    Service-Linked Role Permissions for GuardDuty ........ 17
    Creating a Service-Linked Role for GuardDuty ........... 18
    Editing a Service-Linked Role for GuardDuty .......... 18
    Deleting a Service-Linked Role for GuardDuty ........... 18
  Compliance Validation ............................................... 19
  Resilience ................................................................ 19
  Infrastructure Security ............................................... 19

Setting Up ....................................................................... 21
  Enable Amazon GuardDuty ........................................ 21
  Amazon GuardDuty Free Trial ...................................... 22

Findings ......................................................................... 23
  Locating and Analyzing GuardDuty Findings ............... 23
  Archiving, Downloading, and Providing Feedback on GuardDuty Findings ................................ 25
  Filtering Findings ....................................................... 26
  Suppression Rules ...................................................... 26
    Common Use Cases and Best Practices for Suppression Rules ......................................................... 26
  Severity Levels for GuardDuty Findings ..................... 28
  Generating GuardDuty Sample Findings ................... 28
  Proof of Concept - Automatically Generating Several Common GuardDuty Findings ......................... 29
  GuardDuty Finding Format ......................................... 30

Active Finding Types ................................................... 31
  Backdoor Finding Types ............................................. 31
  Behavior Finding Types ............................................. 34
  CryptoCurrency Finding Types ................................ 35
  PenTest Finding Types ............................................. 35
  Persistence Finding Types ........................................ 36
  Policy Finding Types ............................................... 38
  PrivilegeEscalation Finding Types ............................. 39
  Recon Finding Types ............................................... 39
  ResourceConsumption Finding Types ....................... 43
  Stealth Finding Types ................................................ 43
  Trojan Finding Types ............................................... 45
  Unauthorized Finding Types ...................................... 48

Retired Finding Types .................................................. 53
What Is Amazon GuardDuty?

Amazon GuardDuty is a continuous security monitoring service that analyzes and processes the following data sources: VPC Flow Logs, AWS CloudTrail event logs, and DNS logs. It uses threat intelligence feeds, such as lists of malicious IP addresses and domains, and machine learning to identify unexpected and potentially unauthorized and malicious activity within your AWS environment. This can include issues like escalations of privileges, uses of exposed credentials, or communication with malicious IP addresses, URLs, or domains. For example, GuardDuty can detect compromised EC2 instances serving malware or mining bitcoin. It also monitors AWS account access behavior for signs of compromise, such as unauthorized infrastructure deployments, like instances deployed in a Region that has never been used, or unusual API calls, like a password policy change to reduce password strength.

GuardDuty informs you of the status of your AWS environment by producing security findings that you can view in the GuardDuty console or through Amazon CloudWatch events.

Pricing for GuardDuty

For information about GuardDuty pricing, see Amazon GuardDuty Pricing.

Accessing GuardDuty

You can work with 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.

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.

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 GuardDuty API Reference.
How Amazon GuardDuty Uses Its Data Sources

To detect unauthorized and unexpected activity in your AWS environment, GuardDuty analyzes and processes data from AWS CloudTrail event logs, VPC Flow Logs, and DNS logs. While in transit from these data sources to GuardDuty, all of the log data is encrypted. GuardDuty extracts various fields from these logs for profiling and anomaly detection, and then discards the logs.

The following sections describe the details of how GuardDuty uses each supported data source.

Topics
- AWS CloudTrail event logs (p. 2)
- VPC Flow Logs (p. 3)
- DNS logs (p. 3)

AWS CloudTrail event logs

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 higher-level AWS services. CloudTrail also allows you to identify which users and accounts called AWS APIs for services that support CloudTrail, the source IP address that the calls were made from, and when the calls occurred. For more information, see the AWS CloudTrail User Guide.

You can configure CloudTrail trails to log management events and/or data events. Management events provide insight into management operations that are performed on resources in your AWS account. For example, configuring security (IAM AttachRolePolicy API operations), registering devices (Amazon EC2 CreateDefaultVpc API operations), configuring rules for routing data (Amazon EC2 CreateSubnet API operations), or setting up logging (AWS CloudTrail CreateTrail API operations). Data events provide insight into the resource operations performed on or within a resource. For example, Amazon S3 object-level API activity (GetObject, DeleteObject, and PutObject API operations) or AWS Lambda function execution activity (the Invoke API). For more information, see Logging Data and Management Events for Trails.

Currently, GuardDuty only analyzes CloudTrail management events. If you have CloudTrail configured to log data events, there will be a difference between GuardDuty analysis based on CloudTrail data and the logs that CloudTrail itself is delivering.

Another important detail about GuardDuty's usage of CloudTrail as a data source is the handling and processing of CloudTrail's global events. For most services, events are recorded in the Region where the action occurred. For global services such as AWS IAM, AWS STS, Amazon CloudFront, and Route 53, events are delivered to any trail that includes global services, and are logged as occurring in the US East (N. Virginia) Region. For more information, see About Global Service Events.

GuardDuty processes all events that come into a Region, including global events that CloudTrail sends to all Regions. This allows GuardDuty to maintain user and role profiles in each Region and enables it to accurately detect credentials that are being maliciously used across Regions.

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. If
GuardDuty is not enabled in all supported Regions, its ability to detect activity that involves global services is reduced.

VPC Flow Logs

VPC Flow Logs capture information about the IP traffic going to and from Amazon EC2 network interfaces in your VPC. For more information, see VPC Flow Logs.

When you enable GuardDuty, it immediately starts analyzing your VPC Flow Logs data. It consumes VPC Flow Log events directly from the VPC Flow Logs feature through an independent and duplicative stream of flow logs. This process does not affect any existing flow log configurations that you might have.

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

There is no additional charge for GuardDuty access to flow logs. However, enabling flow logs for retention or use in your account falls under existing pricing. For more information, see Working With Flow Logs.

DNS logs

If you use AWS DNS resolvers for your EC2 instances (the default setting), then GuardDuty can access and process your request and response DNS logs through the internal AWS DNS resolvers. If you are using a 3rd party DNS resolver, for example, OpenDNS or GoogleDNS, or if you set up your own DNS resolvers, then GuardDuty cannot access and process data from this data source.
Concepts and Terminology

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

**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 **master** 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 master account can configure GuardDuty as well as view and manage GuardDuty findings for their own account and all of their member accounts. You can have up to 1000 member accounts in GuardDuty.

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 master 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 Managing Multiple Accounts in Amazon GuardDuty (p. 68).

**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 master account apply to the GuardDuty member accounts. GuardDuty member accounts can't modify suppression rules.

With auto-archive 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 (p. 26)

**Data source**

The origin or location of a set of data. To detect unauthorized and unexpected activity in your AWS environment, GuardDuty analyzes and processes data from AWS CloudTrail event logs, VPC Flow Logs, and DNS logs. For more information, see How Amazon GuardDuty Uses Its Data Sources (p. 2).

**Finding**

A potential security issue discovered by GuardDuty. For more information, see Findings (p. 23).

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 via HTTPS protocol. For more information, see Monitoring GuardDuty Findings with Amazon CloudWatch Events (p. 80).

**Trusted IP list**

A list of whitelisted IP addresses for highly secure communication with your AWS environment. GuardDuty does not generate findings based on trusted IP lists. For more information, see Working with Trusted IP Lists and Threat Lists (p. 64).

**Threat list**

A list of known malicious IP addresses. GuardDuty generates findings based on threat lists. For more information, see Working with Trusted IP Lists and Threat Lists (p. 64).
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 (p. 6)
- Managing Access (p. 7)
- Logging Amazon GuardDuty API Calls with AWS CloudTrail (p. 15)
- Using Service-Linked Roles for Amazon GuardDuty (p. 16)
- Compliance Validation for Amazon GuardDuty (p. 19)
- Resilience in Amazon GuardDuty (p. 19)
- Infrastructure Security in Amazon GuardDuty (p. 19)

Data Protection in Amazon GuardDuty

Amazon GuardDuty conforms to the AWS shared responsibility model, which includes regulations and guidelines for data protection. AWS is responsible for protecting the global infrastructure that runs all the AWS services. AWS maintains control over data hosted on this infrastructure, including the security configuration controls for handling customer content and personal data. AWS customers and APN partners, acting either as data controllers or data processors, are responsible for any personal data that they put in the AWS Cloud.

For data protection purposes, we recommend that you protect AWS account credentials and set up individual user accounts with AWS Identity and Access Management (IAM), so that 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 TLS to communicate with AWS resources.
- Set up API and user activity logging with AWS CloudTrail.
- 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 personal data that is stored in Amazon S3.

We strongly recommend that you never put sensitive identifying information, such as your customers' account numbers, into free-form fields or metadata, such as function names and tags. Any data that you enter into metadata might get picked up for inclusion in diagnostic logs. When you provide a URL to an external server, don't include credentials information in the URL to validate your request to that server.

For more information about data protection, see the AWS Shared Responsibility Model and GDPR blog post on the AWS Security Blog.

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 master keys (CMK).

Encryption in Transit

GuardDuty analyzes log data from other AWS services. GuardDuty 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 AWS services.

Managing Access

Topics
• Permissions Required to Enable GuardDuty (p. 7)
• Using a Service-Linked Role to Delegate Permissions to GuardDuty (p. 8)
• Using IAM Policies to Delegate Access to GuardDuty to IAM Identities (p. 9)

Permissions Required to Enable GuardDuty

This section describes the permissions that various IAM identities (users, groups, and roles) must have in order to initially enable GuardDuty either through the console or programmatically (using the GuardDuty API or the GuardDuty commands in the AWS CLI).

To grant permissions required to enable GuardDuty, attach the following policy to an IAM user, group, or role:

Note
Replace the sample account ID in the example below with your actual AWS account ID.

```json
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [ "guardduty:*"
```
Using a Service-Linked Role to Delegate Permissions to GuardDuty

This section describes the permissions that the GuardDuty service itself requires to function and perform operations on your behalf, such as generating findings.

When you enable GuardDuty (using the GuardDuty console or programmatically through the API operations or AWS CLI commands), it is automatically assigned a service-linked role called AWSServiceRoleForAmazonGuardDuty. A service-linked role is a unique type of IAM role that is linked directly to an AWS service (in this case, GuardDuty). Service-linked roles are predefined by the service and include all the permissions that the service requires to call other AWS services on your behalf. The linked service (in this case, GuardDuty) also defines how you create, modify, and delete a service-linked role. For more information about service-linked roles, see Using Service-Linked Roles.

The AWSServiceRoleForAmazonGuardDuty service-linked role is created automatically when GuardDuty is enabled. It includes the permissions and the trust policies that GuardDuty requires to consume and analyze events directly from AWS CloudTrail, VPC Flow Logs, and DNS logs and generate security findings.

You cannot edit the AWSServiceRoleForAmazonGuardDuty service-linked role. You can view its permissions or delete this service-linked role via the IAM console. To delete the AWSServiceRoleForAmazonGuardDuty service-linked role, you must first disable GuardDuty (p. 79) in all Regions in that AWS account.

To view the permissions attached to AWSServiceRoleForAmazonGuardDuty, choose the View service role permissions button in the Setting/General tab of the GuardDuty console.

The following is the permissions policy document that is attached to the AWSServiceRoleForAmazonGuardDuty service-linked role:

```json
{
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "iam:CreateServiceLinkedRole",
        "iam:PutRolePolicy",
        "iam:DeleteRolePolicy"
      ],
      "Resource": "arn:aws:iam::1234567890123:role/aws-service-role/guardduty.amazonaws.com/AWSServiceRoleForAmazonGuardDuty",
      "Condition": {
        "StringLike": {
          "iam:AWSServiceName": "guardduty.amazonaws.com"
        }
      }
    },
    {
      "Effect": "Allow",
      "Action": [
        "iam:CreateServiceLinkedRole"
      ],
      "Resource": "*"
    }
  ]
}
```
Using IAM Policies to Delegate Access to GuardDuty to IAM Identities

This section describes how to delegate access to GuardDuty to various IAM identities (users, groups, and roles).

By default, access to the GuardDuty resources (detector, trusted IP lists, threat lists, findings, members, master account, and invitations) is restricted to the owner of the AWS account that the resources were created in. If you are the owner, you can choose to grant full or limited access to GuardDuty to the various IAM identities in your account. For more information about creating IAM access policies, see the IAM User Guide.

Topics

- AWS Managed (Predefined) Policies for GuardDuty (p. 9)
- Using a Custom IAM Policy to Grant Full Access to GuardDuty (p. 10)
- Using a Custom IAM Policy to Grant Read-only Access to GuardDuty (p. 11)
- Using a Custom IAM Policy to Deny Access to GuardDuty Findings (p. 11)
- Using a Custom IAM Policy to Limit Access to GuardDuty Resources (p. 12)
- Resources Defined by GuardDuty (p. 14)

AWS Managed (Predefined) Policies for GuardDuty

AWS addresses many common use cases by providing standalone IAM policies that are created and administered by AWS. These managed policies grant necessary permissions for common use cases so that you can avoid having to investigate which permissions are needed. For more information, see AWS Managed Policies in the IAM User Guide.
The following AWS managed policies, which you can attach to users in your account, are specific to GuardDuty:

- **AmazonGuardDutyFullAccess** – provides access to all of GuardDuty functionality. However, when it comes to working with trusted IP lists and threat lists in GuardDuty, this managed policy provides identities with only limited access. More specifically, an identity with the **AmazonGuardDutyFullAccess** managed policy attached 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 uploading, activating, deleting, and updating the location of the lists), make sure that the following actions are present in the permissions policy attached to an IAM user, group, or role:

```json
{
    "Effect": "Allow",
    "Action": [
        "iam:PutRolePolicy",
        "iam:DeleteRolePolicy"
    ],
    "Resource": "arn:aws:iam::123456789123:role/aws-service-role/guardduty.amazonaws.com/AWSServiceRoleForAmazonGuardDuty"
}
```

- **AmazonGuardDutyReadOnlyAccess** – Provides read-only access to GuardDuty.

### Using a Custom IAM Policy to Grant Full Access to GuardDuty

You can use the following custom policy to grant an IAM user, role, or group full access to the GuardDuty console and all GuardDuty operations.

**Note**
Replace the sample account ID in the example below 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::123456789123:role/aws-service-role/guardduty.amazonaws.com/AWSServiceRoleForAmazonGuardDuty",
            "Condition": {
                "StringLike": {
                    "iam:AWSServiceName": "guardduty.amazonaws.com"
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": [
                "iam:PutRolePolicy"
            ],
            "Resource": "*"
        }
    ]
}
```
Using a Custom IAM Policy to Grant Read-only Access to GuardDuty

You can use the following policy to grant an IAM user, role, or group read-only access to GuardDuty:

```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"
      ],
      "Resource": "*"
    }
  ]
}
```

Using a Custom IAM Policy to Deny Access to GuardDuty Findings

You can use the following policy to deny an IAM user, role, 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",
```
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"],
   }
 ]}
```

• 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 Permissions Required to Upload Trusted IP Lists and Threat Lists (p. 65).

```json
{"Version": "2012-10-17",
 "Statement": [
   {
     "Effect": "Allow",
     "Action": ["guardduty:UpdateIPSet"],
   }
 ]}
```

• 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 Permissions Required to Upload Trusted IP Lists and Threat Lists (p. 65).

```json
{
   "Version": "2012-10-17",
   "Statement": [
      {
         "Effect": "Allow",
         "Action": [
            "guardduty:UpdateIPSet",
         ],
         "Resource": "arn:aws:guardduty:us-east-1:012345678910:detector/*/ipset/000000"
      }
   ]
}
```

- This policy allows a user to run the guardduty:UpdateIPSet operation, using the detector ID of 1234567 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 Permissions Required to Upload Trusted IP Lists and Threat Lists (p. 65).

```json
{
   "Version": "2012-10-17",
   "Statement": [
      {
         "Effect": "Allow",
         "Action": [
            "guardduty:UpdateIPSet",
         ],
      }
   ]
}
```

Resources Defined by GuardDuty

<table>
<thead>
<tr>
<th>Resource Type</th>
<th>ARN</th>
</tr>
</thead>
<tbody>
<tr>
<td>detector</td>
<td>arn:${Partition}:guardduty:${Region}:${Account}:detector/${DetectorId}</td>
</tr>
<tr>
<td>ipset</td>
<td>arn:${Partition}:guardduty:${Region}:${Account}:detector/${DetectorId}/ipset/${IPSetId}</td>
</tr>
<tr>
<td>threat intelset</td>
<td>arn:${Partition}:guardduty:${Region}:${Account}:detector/${DetectorId}/threatintelset/${ThreatIntelSetId}</td>
</tr>
<tr>
<td>filter</td>
<td>arn:${Partition}:guardduty:${Region}:${Account}:detector/${DetectorId}/filter/${FilterName}</td>
</tr>
</tbody>
</table>
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 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 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 more.

To learn more 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

All GuardDuty actions are logged by CloudTrail and are documented in the GuardDuty API Reference.

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 or IAM user 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 the CloudTrail userIdentity Element.

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 `CreateIPThreatIntelSet` action.

```
{
  "eventVersion": "1.05",
  "userIdentity": {
    "type": "AssumedRole",
    "principalId": "AIDACKCEVSQ6C2EXAMPLE",
    "arn": "arn:aws:iam::444455556666:user/Alice",
    "accountId": "444455556666",
    "accessKeyId": "AKIAI44QH8DDBEXAMPLE",
    "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": "5ab04b11b0c865eef516eef24e3e7",
      "name": "Example",
      "format": "TXT",
      "activate": false,
      "location": "https://s3.amazonaws.com/bucket.name/file.txt"
    },
    "responseElements": {
      "threatIntelSetId": "1ab200428351c99d859bf61992460d24"
    },
    "requestID": "5f6bf981-7026-11e8-a9fc-5b37d2684c5c",
    "eventType": "AwsApiCall",
    "recipientAccountId": "444455556666"
  }
}
```

From this event information, you can determine that the request was made to create a threat list named `Example` in GuardDuty. You can also see that the request was made by an IAM user named Alice on June 14, 2018.

Using Service-Linked Roles for Amazon GuardDuty

Amazon GuardDuty uses AWS Identity and Access Management (IAM) service-linked roles. A service-linked role 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.

A service-linked role makes setting up GuardDuty easier because you don’t have to manually add the necessary permissions. 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 (p. 90).

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 named AWSServiceRoleForAmazonGuardDuty. This service-linked role allows GuardDuty to retrieve metadata for the EC2 instances in your AWS environment that are involved in potentially suspicious activity. It also allows GuardDuty to include the retrieved EC2 instance metadata in the findings that GuardDuty generates about potentially suspicious activity.

The AWSServiceRoleForAmazonGuardDuty service-linked role trusts the following services to assume the role:

- guardduty.amazonaws.com

The role permissions policy allows GuardDuty to complete the following actions on the specified resources:

- Action: ec2:DescribeInstances
- Action: ec2:DescribeImages
- Resources: arn:aws:iam::*:role/aws-service-role/guardduty.amazonaws.com/AWSServiceRoleForAmazonGuardDuty

You must configure permissions to allow an IAM entity (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 identity that you use GuardDuty with must have the required permissions. To grant the required permissions, attach the following policy to this IAM user, group, or role:

**Note**
Replace the sample account ID in the example below with your actual AWS account ID.

```json
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "guardduty:*"
            ],
            "Resource": null
        }
    ]
}
```
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 AWSServiceRoleForAmazonGuardDuty service-linked role manually using the IAM console, the IAM CLI, or the IAM API.

Important
The service-linked role that is created for the master GuardDuty account doesn't apply to the member GuardDuty accounts.

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
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 (p. 79).
When you disable GuardDuty, the AWSServiceRoleForAmazonGuardDuty is **NOT** automatically deleted. If you then enable GuardDuty again, it'll start using the existing AWSServiceRoleForAmazonGuardDuty.

**To manually delete the service-linked role using IAM**

Use the IAM console, the IAM 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.

### Compliance Validation for Amazon GuardDuty

Third-party auditors assess the security and compliance of GuardDuty as part of multiple AWS compliance programs. These include SOC, PCI, FedRAMP, HIPAA, and others.

For a list of AWS services in scope of specific compliance programs, see AWS Services in Scope by Compliance Program. 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 GuardDuty 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 and Compliance Quick Start Guides** – These deployment guides discuss architectural considerations and provide steps for deploying security- and compliance-focused baseline environments on AWS.
- **Architecting for HIPAA Security and Compliance Whitepaper** – This whitepaper describes how companies can use AWS to create HIPAA-compliant applications.
- **AWS Compliance Resources** – This collection of workbooks and guides might apply to your industry and location.
- **Evaluating Resources with Rules** in the *AWS Config Developer Guide* – AWS Config; 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 that helps you check your compliance with security industry standards and best practices. GuardDuty integrates with Security Hub to consolidate findings.

### 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, GuardDuty is protected by the AWS global network security procedures that are described in the Amazon Web Services: Overview of Security Processes whitepaper.
You use AWS published API calls to access GuardDuty through the network. Clients must support Transport Layer Security (TLS) 1.0 or later. We recommend TLS 1.2 or later. Clients must also support cipher suites with perfect forward secrecy (PFS) such as Ephemeral Diffie-Hellman (DHE) or Elliptic Curve Ephemeral Diffie-Hellman (ECDHE). Most modern systems such as Java 7 and later support these modes.

Additionally, requests must be signed 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.
Setting Up GuardDuty

You must have an AWS account in order to enable Amazon GuardDuty. If you don’t have an account, use the following procedure to create one.

To sign up for AWS

2. Follow the online instructions.

   Part of the sign-up procedure involves receiving a phone call and entering a verification code on the phone keypad.

Topics

• Enable Amazon GuardDuty (p. 21)
• Amazon GuardDuty Free Trial (p. 22)

Enable Amazon GuardDuty

To use GuardDuty, you must first enable it. Use the following procedure to enable GuardDuty.

1. The IAM identity (user, role, group) that you use to enable GuardDuty must have the required permissions. To grant the permissions required to enable GuardDuty, attach the following policy to an IAM user, group, or role:

   Note
   Replace the sample account ID in the example below 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"
                }
            }
        }
    ]
}
```
2. Use the credentials of the IAM identity from step 1 to sign in to the GuardDuty console. When you open the GuardDuty console for the first time, choose **Get Started**, and then choose **Enable GuardDuty**.

Note the following about enabling GuardDuty:

- GuardDuty is assigned a service-linked role called **AWSServiceRoleForAmazonGuardDuty**. This service-linked role allows GuardDuty to retrieve metadata for the EC2 instances in your AWS environment that are involved in potentially suspicious activity. It also allows GuardDuty to include the retrieved EC2 instance metadata in the findings that GuardDuty generates about potentially suspicious activity. To view the details of **AWSServiceRoleForAmazonGuardDuty**, on the **Welcome to GuardDuty** page, choose **View service role permissions**. For more information, see Using a Service-Linked Role to Delegate Permissions to GuardDuty (p. 8). For more information about service-linked roles, see Using Service-Linked Roles.

- After you enable GuardDuty, it immediately begins pulling and analyzing independent streams of data from AWS CloudTrail, VPC Flow Logs, and DNS logs to generate security findings. Because GuardDuty only consumes this data for purposes of determining if there are any findings, GuardDuty doesn't manage AWS CloudTrail, VPC Flow Logs, and DNS logs for you or make their events and logs available to you. If you have enabled these services independent of GuardDuty, you will continue to have the option to configure the settings of these data sources through their respective consoles or APIs. For more information about the data sources that GuardDuty integrates with, see What is AWS CloudTrail? and Working With Flow Logs.

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.

- You can disable GuardDuty at any time to stop it from processing and analyzing AWS CloudTrail events, VPC Flow Logs, and DNS logs. For more information, see Suspending or Disabling GuardDuty (p. 79).

## Amazon GuardDuty Free Trial

When you enable GuardDuty for the first time, your AWS account is automatically enrolled in a 30-day GuardDuty free trial. You can view the details of your GuardDuty free trial in the **Free trial** page of the GuardDuty console (choose **Free trial / Details** in the navigation pane). The details include your current position on the free trial timeline and the estimated daily cost for using GuardDuty after your free trial ends. This estimate is based on the logs that GuardDuty processes and analyzes daily during the free trial.

This estimated daily cost does NOT project charges for using GuardDuty in all of the AWS accounts and Regions where you enabled GuardDuty. The daily cost estimate is based only on the GuardDuty usage in the AWS account and the AWS Region to which you're currently signed in.

You will not be charged for using GuardDuty until your free trial ends. For more information about GuardDuty pricing, see Amazon GuardDuty Pricing.
Findings

GuardDuty generates findings when 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 GuardDuty CLI or API operations. You can also view your GuardDuty findings through Amazon CloudWatch events. For more information, see Monitoring GuardDuty Findings with Amazon CloudWatch Events (p. 80).

This topic describes the following information:

Topics

- Locating and Analyzing GuardDuty Findings (p. 23)
- Archiving, Downloading, and Providing Feedback on GuardDuty Findings (p. 25)
- Filtering Findings (p. 26)
- Suppression Rules (p. 26)
- Severity Levels for GuardDuty Findings (p. 28)
- Generating GuardDuty Sample Findings (p. 28)
- Proof of Concept - Automatically Generating Several Common GuardDuty Findings (p. 29)
- GuardDuty Finding Format (p. 30)
- Active Finding Types (p. 31)
- Retired Finding Types (p. 53)
- Remediating Security Issues Discovered by GuardDuty (p. 55)

Locating and Analyzing GuardDuty Findings

Use the following procedure to view and analyze your GuardDuty findings.

2. Choose Findings and then choose a specific finding to view its details.

   A details pane appears where you can view the following information:

   - A finding's summary section that includes the following information:
     - Finding type – a concise yet readable description of the potential security issue. For more information, see GuardDuty Finding Format (p. 30).
     - Severity – a finding's assigned severity level of either High, Medium, or Low. For more information, see Severity Levels for GuardDuty Findings (p. 28).
     - Region – the AWS Region in which the finding was generated.

       **Note**

       For more information about supported Regions, see Regions and Endpoints (p. 90)

   - Count – the number of times GuardDuty generated the finding after you enabled GuardDuty in your AWS account.
   - Account ID – the ID of the AWS account in which the activity took place that prompted GuardDuty to generate this finding.
   - Resource ID – the ID of the AWS resource against which the activity took place that prompted GuardDuty to generate this finding.
   - 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.
• **Last seen** – the time at which the activity took place that prompted GuardDuty to generate this finding.

  **Note**
  Findings' time stamps in the GuardDuty console appear in your local time zone, while JSON exports and CLI outputs display timestamps in UTC.

• A finding's **Resource affected** section that includes the following information:
  
  • **Resource role** – a value that usually is set to **Target** because the affected resource can be a potential target of an attack.
  
  • **Resource type** – the type of the affected resource. This value is either **AccessKey** or **Instance**. Currently, supported finding types highlight potentially malicious activity against either EC2 instances or AWS credentials. For more information, see [Remediating Security Issues Discovered by GuardDuty](p. 55).
  
  • **Instance ID** – the ID of the EC2 instance involved in the activity that prompted GuardDuty to generate the finding.
  
  • **Outpost ARN** – The Amazon Resource Name (ARN) of the AWS Outpost. Only applicable to AWS Outposts instances. For more information, see [What is AWS Outposts?](p. 55).
  
  • **Port** – the port number for the connection used during the activity that prompted GuardDuty to generate the finding.
  
  • **Access key ID** – 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](p. 55).
  
  • **User name** – The name of the user engaged in the activity that prompted GuardDuty to generate the finding.

• A finding's **Action** section that can include the following information:
  
  • **Action type** – the finding activity type. This value can be one of the following: NETWORK_CONNECTION, AWS_API_CALL, PORT_PROBE, or DNS_REQUEST. NETWORK_CONNECTION indicates that network traffic was exchanged between the identified EC2 instance and the remote host. AWS_API_CALL indicates that an AWS API was invoked. DNS_REQUEST indicates that the identified EC2 instance queried a domain name. PORT_PROBE indicates that a remote host probed the identified EC2 instance on multiple open ports.
  
  • **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 CloudTrail. For more information, see [Non-API Events Captured by CloudTrail](p. 55).
  
  • **Service name** – the name of the AWS service (GuardDuty) that generated the finding.
  
  • **Connection direction** – the network connection direction observed in the activity that prompted GuardDuty to generate the finding. The values can be INBOUND, OUTBOUND, and UNKNOWN. 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 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.
Archiving, Downloading, and Providing Feedback on GuardDuty Findings

Use the following procedure to archive your findings or mark them as current and to provide feedback for your GuardDuty findings.

1. To archive or download a finding, choose it from the list of your findings and then choose the Actions menu. Then choose Archive or Export. When you Export a finding, you can see its full JSON document.

   **Note**
   Currently in GuardDuty, users from GuardDuty member accounts can't archive findings.

   **Note**
   If, and only if, the confidence level of a GuardDuty finding is set to 0, the Confidence field is displayed in the full finding JSON. The presence of the Confidence field set to 0 indicates that this GuardDuty finding is a false positive.

2. To provide feedback by marking the finding useful or not useful, choose it from the list of your findings and then choose the thumbs up or thumbs down icons.

3. To view your archived or current findings, choose the filter icon above the list of findings and then check either the Archived or the Current checkbox.

   **Important**
   If you archive a finding manually using the procedure above, all subsequent occurrences of this finding (generated after the archiving is complete) are added to the list of your current findings. To never see this finding in your current list, you can auto-archive it. For more information, see Filtering Findings (p. 26).
Filtering Findings

Use the following procedure to create filters for your GuardDuty findings.

To filter findings

1. Choose the Add filter criteria bar above the displayed list of your GuardDuty findings.
2. In the expanded list of attributes, select the attributes that you want to specify as the criteria for your filter. For example, Account ID and/or Action type. You can specify one attribute or a maximum of 50 attributes as the criteria for a particular filter.

   For the complete list of attributes that you can specify as filter criteria, see the details of the findingCriteria parameter of the CreateFilter operation.
3. In the displayed text field, specify a value for each selected attribute and then select Apply.

   Note
   When you use the ‘equal to’ or ‘not equal to’ condition to filter on an attribute value, such as Account ID, you can specify a maximum of 50 values. After you apply a filter, you can convert the filter to exclude findings that match the filter by choosing the black dot to the left side of the filter name. This effectively creates a “not equals” filter for the selected attribute.
4. To save the specified attributes and their values (filter criteria) as a filter, select Save. Provide the filter name and description, and then choose Done.

Suppression Rules

A suppression rule is a filter used to automatically archive new findings. After you create a suppression rule, new findings that match the criteria defined in the rule are automatically archived. You can use an existing filter to create a suppression rule, or define a new filter for the suppression rule as you create it.

To create a suppression rule

1. On the GuardDuty Findings page, choose Suppress findings.
2. Do one of the following:
   • To use an existing filter for the rule, choose the filter from the Saved rules drop-down list.
   • Add filter criteria to create a filter for the suppression rule. To save the filter for use later, choose Save at the end of the filter criteria field.
3. Enter a name and a description for the suppression rule.
4. Choose Save.

Common Use Cases and Best Practices for Suppression Rules

One of the most important steps when setting up GuardDuty is to fine-tune the service to your specific AWS environment. You can achieve this by using GuardDuty finding suppression rules. Suppression rules result in reduced noise levels of generated findings, letting you focus on findings that may indicate a security threat. Noisy findings can represent either activity that is expected for your AWS environment or activity that is not a threat that you intend to act on. When a finding matches a suppression rule, it is still generated and stored in GuardDuty for 90-days. However, it is automatically marked as “archived”. Accordingly, it does not appear in the “current findings” section of console, clearing away the noise and enabling you to focus on findings that are of higher security value and represent unexpected behavior for you AWS environment.
You can still view suppressed findings in the archived section of the findings table. Suppressed findings are not sent to AWS Security Hub, Amazon S3, CloudWatch Events. This helps lower the noise level if you consume GuardDuty findings via Security Hub or a third-party SIEM, alerting and ticketing applications. As the findings are still generated, we recommend periodically viewing the archived findings to validate that the rules are configured properly to only match findings that you wish to suppress. 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. Following are common use cases for applying suppression rules:

**UnauthorizedAccess:IAMUser/InstanceCredentialExfiltration – Using 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**

The instance credential exfiltration finding informs you of attempts to run AWS API operations from a host outside of EC2, using temporary AWS credentials that were created on an EC2 instance in your AWS account. This finding may indicate that your EC2 instance is compromised, and the temporary credentials from this instance might have been exfiltrated to a remote host outside of AWS. If this behavior is unexpected this can indicate a high severity security incident and should be addressed immediately. However, this finding is generated if VPC networking is configured to route Internet traffic such that it egresses from an on-premise gateway rather than from a VPC Internet Gateway (IGW). Common configurations that result in this include using AWS Direct Connect, AWS Outposts, or VPC VPN connections. To suppress this expected behavior it is recommended that you create a suppression rule using the **Outpost Arn** attribute if the findings are for an AWS Outpost. For other configurations create a filter based on two criteria: The first criteria should use the **Finding type** attribute with a value of `UnauthorizedAccess:IAMUser/InstanceCredentialExfiltration`. The second filter criteria should use either the IP address or ASN of your on-premises internet gateway. For IP-based filters, use the **API caller IPv4 address** attribute, and for ASN based filters use either the **API caller ASN name** attribute or the **API caller ASN ID** attribute.

**Recon:EC2/Portscan – Using a suppression rule to automatically archive findings when using a vulnerability assessment application**

The Portscan finding informs you that there is an EC2 instance in your AWS environment that 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 that the machine is running and to identify its operating system. This may indicate that your EC2 instance might be compromised. However, this finding is often triggered when vulnerability assessment applications are deployed on EC2 Instances in your environment, as these applications conduct portscans 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 what criteria is identifiable with the instances that host these tools.

**UnauthorizedAccess:EC2/SSHBruteForce – Using a suppression rule to automatically archive findings when it is targeted to bastion instances**

The SSH Brute Force finding informs you that an EC2 instance in your AWS environment was involved in an SSH brute force attempt aimed at obtaining passwords to SSH services on Linux-based systems. This can indicate unauthorized access to your AWS resources. However, 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 what criteria is identifiable with the instances that host these tools.
Recon:EC2/PortProbeUnprotectedPort – Using a suppression rule to automatically archive findings when it is targeted to intentionally exposed instances

The Port Probe finding informs you that a port on an EC2 instance in your AWS environment is not blocked by a security group, access control list (ACL), or an on-host firewall (for example, Linux IPTables), and known scanners on the internet are actively probing it. However, 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 what criteria is identifiable with the instances that host these tools.

Severity Levels for GuardDuty Findings

Each GuardDuty finding has an assigned severity level and value that reduces the need to prioritize one finding over another and can help you determine your response to a potential security issue that is highlighted by a finding. The value of the severity can fall anywhere within the 0.1 to 8.9 range.

Note
Values 0 and 9.0 to 10.0 are currently reserved for future use.

The following are the currently defined severity levels and values for the GuardDuty findings:

- **High** (the value of the severity parameter in the GetFindings response falls within the 7.0 to 8.9 range) – indicates that the resource in question (an EC2 instance or a set of IAM user credentials) is compromised and is actively being used for unauthorized purposes. We recommend that you treat this security issue as a priority and take immediate remediation steps. For example, clean up your EC2 instance or terminate it, or rotate the IAM credentials.

- **Medium** (the value of the severity parameter in the GetFindings response falls within the 4.0 to 6.9 range) – indicates suspicious activity, for example, a large amount of traffic being returned to a remote host that is hiding behind the Tor network, or activity that deviates from normally observed behavior. We recommend that you investigate the implicated resource at your earliest convenience. Here are some of the possible remediation steps:
  - 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 panel 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** (the value of the severity parameter in the GetFindings response falls within the 0.1 to 3.9 range) – indicates attempted suspicious activity that did not compromise your network, for example, a port scan or a failed intrusion attempt. There is no immediate recommended action, but it is worth making note of this information as it may indicate someone is looking for weak points in your network.

Generating GuardDuty Sample Findings

Sample findings can help you visualize and analyze the various finding types that GuardDuty generates. When you generate sample findings, GuardDuty populates your current findings list with one sample finding for each supported finding type. For more information about GuardDuty finding types, see Active Finding Types (p. 31).
Use the following procedure to generate sample findings.

2. In the navigation pane, under Settings, choose General.
3. On the Settings page, under Sample findings, choose Generate sample findings.
4. In the navigation pane, under Findings, choose Current. The sample findings are displayed on the Current findings page. The title of sample findings always begins with [SAMPLE]. Choose a specific sample finding to view its details.

Proof of Concept - Automatically Generating Several Common GuardDuty Findings

You can use the following scripts to automatically generate several common GuardDuty findings. **guardduty-tester.template** uses AWS CloudFormation to create an isolated environment with a bastion host, a tester EC2 instance that you can ssh into, and two target EC2 instances. Then you can run **guardduty_tester.sh** that starts interaction between the tester EC2 instance and the target Windows EC2 instance and the target Linux EC2 instance to simulate five types of common attacks that GuardDuty is built to detect and notify you about with generated findings.

1. As a prerequisite, you must enable GuardDuty in the same account and Region where you want to run the **guardduty-tester.template** and **guardduty_tester.sh**. For more information about enabling GuardDuty, see Setting Up GuardDuty (p. 21).

   You must also generate a new or use an existing EC2 key pair in each Region where you want to run these scripts. This EC2 key pair is used as a parameter in the **guardduty-tester.template** script that you use to create a new CloudFormation stack. For more information about generating key pairs, see Amazon EC2 Key Pairs.

2. Create a new CloudFormation stack using **guardduty-tester.template**. For detailed instructions about creating a stack, see Creating a Stack. Before you run **guardduty-tester.template**, modify it with values for the following parameters: Stack Name to identify your new stack, Availability Zone where you want to run the stack, and Key Pair that you can use to launch the EC2 instances. Then you can use the corresponding private key to SSH into the EC2 instances.

   **guardduty-tester.template** takes around 10 minutes to run and complete. It creates your environment and copies **guardduty_tester.sh** onto your tester EC2 instance.

3. In the AWS CloudFormation console, choose the checkbox next to your new running AWS CloudFormation stack. In the displayed set of tabs, select the Output tab. Note the IP addresses assigned to the bastion host and the tester EC2 instance. You need both of these IP addresses in order to ssh into the tester EC2 instance.

4. Create the following entry in your ~/.ssh/config file to login to your instance through the bastion host:

   ```
   Host bastion
   HostName {Elastic IP Address of Bastion}
   User ec2-user
   IdentityFile ~/.ssh/{your-ssh-key.pem}
   Host tester
   ForwardAgent yes
   HostName {Local IP Address of RedTeam Instance}
   User ec2-user
   IdentityFile ~/.ssh/{your-ssh-key.pem}
   ProxyCommand ssh bastion nc %h %p
   ```
Now you can call $ ssh tester to login to your target EC2 instance. For more information about configuring and connecting to EC2 instances through bastion hosts, see https://aws.amazon.com/blogs/security/securely-connect-to-linux-instances-running-in-a-private-amazon-vpc/.

5. Once connected to the tester EC2 instance, run `guardduty_tester.sh` to initiate interaction between your tester and target EC2 instances, simulate attacks, and generate GuardDuty Findings.

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 finding details (p. 23) include information about what happened, which AWS resources were involved in the suspicious activity, when this activity took place, and other information.

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 the various types of findings that it generates:

```
ThreatPurpose:ResourceTypeAffected/ThreatFamilyName.ThreatFamilyVariant!Artifact
```

This is what each part of the format represents:

- **ThreatPurpose** - describes the primary purpose of a threat or a potential attack. In the current release of GuardDuty, ThreatPurpose can have the following values:
  - **Backdoor** - this value indicates that the attack has compromised an AWS resource and 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 is detecting activity or activity patterns that are different from the established baseline for a particular AWS resource.
  - **Cryptocurrency** - this value indicates that GuardDuty is detecting software that is associated with cryptocurrencies (for example, Bitcoin).
  - **Pentest** - Sometimes owners of AWS resources or their authorized representatives intentionally run tests against AWS applications to find vulnerabilities, like 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 attackers. However, some of the tools used by authorized pen testers are freely available, and therefore can be used by unauthorized users or attackers 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 and that it is similar to the activity generated by known pen testing tools. Therefore, it can be a potential attack.
  - **Persistence** - this value indicates that a principal in your AWS environment is exhibiting behavior that is different from the established baseline. For example, this principal has no prior history of updating network configuration settings, or updating policies or permissions attached to AWS users or resources.
  - **Policy** - this value indicates that your AWS account is exhibiting behavior that goes against recommended security best practices.
  - **PrivilegeEscalation** - this value informs you that a specific principal in your AWS environment is exhibiting behavior that can be indicative of a privilege escalation attack.
  - **Recon** - this value indicates that a reconnaissance attack is underway, scoping out vulnerabilities in your AWS environment by probing ports, listing users, database tables, and so on.
Active Finding Types

GuardDuty generates finding types of the following threat purposes:

Finding Types
- Backdoor Finding Types (p. 31)
- Behavior Finding Types (p. 34)
- CryptoCurrency Finding Types (p. 35)
- PenTest Finding Types (p. 35)
- Persistence Finding Types (p. 36)
- Policy Finding Types (p. 38)
- PrivilegeEscalation Finding Types (p. 39)
- Recon Finding Types (p. 39)
- ResourceConsumption Finding Types (p. 43)
- Stealth Finding Types (p. 43)
- Trojan Finding Types (p. 45)
- Unauthorized Finding Types (p. 48)

Backdoor Finding Types

This section covers the active Backdoor threat purpose finding types. For information about important changes to the GuardDuty finding types, including newly added or retired finding types, see Document History for Amazon GuardDuty (p. 91).
Important
The default severity value of a finding type is subject to change based on various criteria when the finding is generated.

Topics
- Backdoor:EC2/Spambot (p. 32)
- Backdoor:EC2/C&CActivity.B!DNS (p. 32)
- Backdoor:EC2/DenialOfService.Tcp (p. 32)
- Backdoor:EC2/DenialOfService.Udp (p. 33)
- Backdoor:EC2/DenialOfService.Dns (p. 33)
- Backdoor:EC2/DenialOfService.UdpOnTcpPorts (p. 33)
- Backdoor:EC2/DenialOfService.UnusualProtocol (p. 34)

Backdoor:EC2/Spambot

Finding description
EC2 instance is exhibiting unusual behavior by communicating with a remote host on port 25.

This finding informs you that an 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. Your EC2 instance might be compromised and sending out spam. For more information, see Remediating a Compromised EC2 Instance (p. 55).

Default severity: Medium

Backdoor:EC2/C&CActivity.B!DNS

Finding description
EC2 instance is querying a domain name that is associated with a known command and control server.

This finding informs you that there is an EC2 instance in your AWS environment that is querying a domain name associated with a known command and control (C&C) server. Your EC2 instance might be 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 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. For more information, see Remediating a Compromised EC2 Instance (p. 55).

Note
To test how GuardDuty's generates this finding type you can make a DNS request against a test domain guardian.com2activityb.com.

Default severity: High

Backdoor:EC2/DenialOfService.Tcp

Finding description
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 TCP protocol.
This finding informs you that there is an EC2 instance in your AWS environment that is generating a large volume of outbound TCP traffic. This may indicate that the instance is compromised and is being used to perform Denial-of-Service (DoS) attacks using TCP protocol. This finding detects DoS attacks only against publicly routable IP addresses, which are primary targets of DoS attacks. For more information, see Remediating a Compromised EC2 Instance (p. 55).

**Default severity: High**

**Backdoor:EC2/DenialOfService.Udp**

**Finding description**

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.

This finding informs you that there is an EC2 instance in your AWS environment that is generating a large volume of outbound UDP traffic. This may indicate that the instance is compromised and is being used to perform Denial-of-Service (DoS) attacks using UDP protocol. This finding detects DoS attacks only against publicly routable IP addresses, which are primary targets of DoS attacks. For more information, see Remediating a Compromised EC2 Instance (p. 55).

**Default severity: High**

**Backdoor:EC2/DenialOfService.Dns**

**Finding description**

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.

This finding informs you that there is an EC2 instance in your AWS environment that is generating a large volume of outbound DNS traffic. This may indicate that the instance is compromised and is being used to perform Denial-of-Service (DoS) attacks using DNS protocol. This finding detects DoS attacks only against publicly routable IP addresses, which are primary targets of DoS attacks. For more information, see Remediating a Compromised EC2 Instance (p. 55).

**Default severity: High**

**Backdoor:EC2/DenialOfService.UdpOnTcpPorts**

**Finding description**

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.

This finding informs you that there is an EC2 instance in your AWS environment that 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 instance is compromised and is being used to perform a Denial-of-Service (DoS) attacks using UDP protocol on a TCP port. This finding detects DoS attacks only against publicly routable IP addresses, which are primary targets of DoS attacks. For more information, see Remediating a Compromised EC2 Instance (p. 55).
Behavior Finding Types

This section covers the active Behavior threat purpose finding types. For information about important changes to the GuardDuty finding types, including newly added or retired finding types, see Document History for Amazon GuardDuty (p. 91).

Important
The default severity value of a finding type is subject to change based on various criteria when the finding is generated.

Topics
- Behavior:EC2/NetworkPortUnusual (p. 34)
- Behavior:EC2/TrafficVolumeUnusual (p. 34)

Default severity: High

Behavior:EC2/NetworkPortUnusual

Finding description

An EC2 instance is communicating with a remote host on an unusual server port.

This finding informs you that an 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. Your EC2 instance might be compromised. For more information, see Remediating a Compromised EC2 Instance (p. 55).

Default severity: Medium

Behavior:EC2/TrafficVolumeUnusual

Finding description

An EC2 instance is generating unusually large amounts of network traffic to a remote host.

This finding informs you that an 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. Your EC2 instance might be compromised. For more information, see Remediating a Compromised EC2 Instance (p. 55).
Default severity: Medium

CryptoCurrency Finding Types

This section covers the active CryptoCurrency threat purpose finding types. For information about important changes to the GuardDuty finding types, including newly added or retired finding types, see Document History for Amazon GuardDuty (p. 91).

Important
The default severity value of a finding type is subject to change based on various criteria when the finding is generated.

Topics
- CryptoCurrency:EC2/BitcoinTool.B!DNS (p. 35)
- CryptoCurrency:EC2/BitcoinTool.B (p. 35)

CryptoCurrency:EC2/BitcoinTool.B!DNS

Finding description
EC2 instance is querying a domain name that is associated with cryptocurrency-related activity.

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. Bitcoin is a worldwide cryptocurrency and digital payment system. Besides being created as a reward for Bitcoin mining, Bitcoin can be exchanged for other currencies, products, and services. Unless you use this EC2 instance to mine or manage cryptocurrency or your EC2 instance is involved in blockchain activity, your EC2 instance might be compromised. For more information, see Remediating a Compromised EC2 Instance (p. 55).

Default severity: High

CryptoCurrency:EC2/BitcoinTool.B

Finding description
EC2 instance is querying an IP address that is associated with cryptocurrency-related activity.

This finding informs you that an 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. Besides being created as a reward for Bitcoin mining, Bitcoin can be exchanged for other currencies, products, and services. Unless you use this EC2 instance to mine or manage cryptocurrency or your EC2 instance is involved in blockchain activity, your EC2 instance might be compromised. For more information, see Remediating a Compromised EC2 Instance. For more information, see Remediating a Compromised EC2 Instance (p. 55).

Default severity: High

PenTest Finding Types

This section covers the active PenTest threat purpose finding types. For information about important changes to the GuardDuty finding types, including newly added or retired finding types, see Document History for Amazon GuardDuty (p. 91).

Important
The default severity value of a finding type is subject to change based on various criteria when the finding is generated.
PenTest:IAMUser/KaliLinux

Finding description

An API was invoked from a Kali Linux EC2 instance.

This finding informs you that a machine running Kali Linux is making 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. For more information, see Remediating Compromised AWS Credentials (p. 55).

Default severity: Medium

PenTest:IAMUser/ParrotLinux

Finding description

An API was invoked from a Parrot Security Linux EC2 instance.

This finding informs you that a machine running Parrot Security Linux is making 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. For more information, see Remediating Compromised AWS Credentials (p. 55).

Default severity: Medium

PenTest:IAMUser/PentooLinux

Finding description

An API was invoked from a Pentoo Linux EC2 instance.

This finding informs you that a machine running Pentoo Linux is making 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. For more information, see Remediating Compromised AWS Credentials (p. 55).

Default severity: Medium

Persistence Finding Types

This section covers the active Persistence threat purpose finding types. For information about important changes to the GuardDuty finding types, including newly added or retired finding types, see Document History for Amazon GuardDuty (p. 91).
Important
The default severity value of a finding type is subject to change based on various criteria when the finding is generated.

Topics
- Persistence:IAMUser/NetworkPermissions (p. 37)
- Persistence:IAMUser/ResourcePermissions (p. 37)
- Persistence:IAMUser/UserPermissions (p. 38)

Persistence:IAMUser/NetworkPermissions
Finding description

A principal invoked an API commonly used to change the network access permissions for security groups, routes, and ACLs in your AWS account.

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 invoking this API. Your credentials might be compromised. For more information, see Remediating Compromised AWS Credentials (p. 55).

This finding is triggered when network configuration settings are changed under suspicious circumstances. For example, if a principal in your AWS environment with no prior history of doing so, invoked the CreateSecurityGroup API. Attackers often attempt to change security groups, allowing certain inbound traffic on various ports in order to improve their ability to access the bot they might have planted on your EC2 instance.

Default severity: Medium

This finding’s default severity is Medium. However, if the API is invoked using temporary AWS credentials that are created on an Amazon EC2 instance, the finding’s severity is High.

Persistence:IAMUser/ResourcePermissions
Finding description

A principal invoked an API commonly used to change the security access policies of various resources in your AWS account.

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 invoking this API. Your credentials might be compromised. For more information, see Remediating Compromised AWS Credentials (p. 55).

This finding is triggered when a change is detected to policies or permissions attached to AWS resources. For example, if a principal in your AWS environment with no prior history of doing so, invoked the PutBucketPolicy API. Some services, for example, 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, granting themselves future access to that resource.

Default severity: Medium

This finding’s default severity is Medium. However, if the API is invoked using temporary AWS credentials that are created on an Amazon EC2 instance, the finding’s severity is High.
Persistence: IAMUser/UserPermissions

Finding description

A principal invoked an API commonly used to add, modify, or delete IAM users, groups or policies in your AWS account.

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 invoking this API. Your credentials might be compromised. For more information, see Remediating Compromised AWS Credentials (p. 55).

This finding is triggered by suspicious changes to the user-related permissions in your AWS environment. For example, if a principal in your AWS environment with no prior history of doing so, invoked the AttachUserPolicy API. In an effort to maximize their ability to access the account even after they have been discovered, attackers can use stolen credentials to create new users, add access policies to existing users, create access keys, etc. The owner of the account might notice that a particular IAM user or password was stolen and delete it from the account, but might not delete other users that were created by the fraudulently created admin principal, leaving their AWS account still accessible to the attacker.

Default severity: Medium

This finding’s default severity is Medium. However, if the API is invoked using temporary AWS credentials that are created on an Amazon EC2 instance, the finding’s severity is High.

Policy Finding Types

This section covers the active Policy threat purpose finding types. For information about important changes to the GuardDuty finding types, including newly added or retired finding types, see Document History for Amazon GuardDuty (p. 91).

Important

The default severity value of a finding type is subject to change based on various criteria when the finding is generated.

Topics

- Policy: IAMUser/S3BlockPublicAccessDisabled (p. 38)
- Policy: IAMUser/RootCredentialUsage (p. 39)

Policy: IAMUser/S3BlockPublicAccessDisabled

Finding description

Amazon S3 block public access was disabled for a bucket

This finding informs you that Amazon S3 block public access was disabled for the S3 bucket referenced under AffectedResources. When enabled, S3 block public access settings are used to filter the policies or ACLs applied to the bucket to prevent inadvertent public exposure of data. When S3 block public access is disabled, all policies or ACLs applied to the bucket are used to determine who can access it, including any public access. Typically, S3 block public access is turned off to allow public access to a bucket or the objects in the bucket. Because S3 block public access is now disabled for this bucket, any policies or ACLs applied to the bucket are now controlling access to the bucket. 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. To learn more, see Using Amazon S3 Block Public Access

If you did not disable S3 block public access for the bucket, it may indicate that your credentials are compromised. To learn more, Remediating Compromised AWS Credentials (p. 55).
Default severity: Low

**Policy:IAMUser/RootCredentialUsage**

Finding description

An API was invoked using root credentials.

This finding informs you that the root credentials of your AWS account are being used to make requests to AWS services. It is recommended that you do NOT use your account's root credentials to access AWS services. Instead, you can access AWS services using least-privilege temporary credentials from AWS Security Token Service (STS). For situations where STS is not supported, you can use IAM user credentials. For more information, see IAM Best Practices. For more information, see Remediating Compromised AWS Credentials (p. 55).

Default severity: Low

**PrivilegeEscalation Finding Types**

This section covers the active PrivilegeEscalation threat purpose finding types. For information about important changes to the GuardDuty finding types, including newly added or retired finding types, see Document History for Amazon GuardDuty (p. 91).

**Important**

The default severity value of a finding type is subject to change based on various criteria when the finding is generated.

Topics

- PrivilegeEscalation:IAMUser/AdministrativePermissions (p. 39)

**PrivilegeEscalation:IAMUser/AdministrativePermissions**

Finding description

A principal has attempted to assign a highly permissive policy to themselves. This behavior is associated with a privilege escalation attack.

This finding informs you that a specific principal in your AWS environment is exhibiting behavior that can be indicative of a privilege escalation attack. This finding is triggered when a 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. In an effort to maximize their ability to access your account, even after they have been discovered, attackers can use stolen credentials or a misconfigured role to escalate their privileges and then create new users, add access policies to existing users, create access keys, etc. For more information, see Remediating Compromised AWS Credentials (p. 55).

Default severity: Low

This finding's severity is Low if the attempt at privilege escalation was unsuccessful, and Medium if the attempt at privilege escalation was successful.

**Recon Finding Types**

This section covers the active Recon threat purpose finding types. For information about important changes to the GuardDuty finding types, including newly added or retired finding types, see Document History for Amazon GuardDuty (p. 91).
Important
The default severity value of a finding type is subject to change based on various criteria when the finding is generated.

Topics
- Recon:EC2/PortProbeUnprotectedPort (p. 40)
- Recon:EC2/PortProbeEMRUnprotectedPort (p. 40)
- Recon:IAMUser/TorIPCaller (p. 41)
- Recon:IAMUser/MaliciousIPCaller Custom (p. 41)
- Recon:IAMUser/MaliciousIPCaller (p. 41)
- Recon:EC2/Portscan (p. 41)
- Recon:IAMUser/NetworkPermissions (p. 42)
- Recon:IAMUser/ResourcePermissions (p. 42)
- Recon:IAMUser/UserPermissions (p. 42)

Recon:EC2/PortProbeUnprotectedPort

Finding description

EC2 instance has an unprotected port that is being probed by a known malicious host.

This finding informs you that a port on an EC2 instance in your AWS environment is not blocked by a security group, access control list (ACL), or on-host firewall (e.g., Linux IPTables), and known scanners on the internet are actively probing it. If the identified unprotected port is 22 or 3389 and you often connect to this EC2 instance by using SSH/RDP and therefore can’t block access to either of these ports, 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.

For more information, see Remediating a Compromised EC2 Instance (p. 55).

Default severity: Low

Note
This finding’s default severity is Low. However, if the port being probed is used by Elasticsearch (9200 or 9300), the finding’s severity is High.

Recon:EC2/PortProbeEMRUnprotectedPort

Finding description

This finding informs you that an EMR-related sensitive port on an EC2 instance that is part of an EMR cluster in your AWS environment is not blocked by a security group, access control list (ACL), or on-host firewall (e.g., Linux IPTables), and known scanners on the internet are actively probing it. Ports that can trigger this finding, such as the port 8088 (YARN Web UI port), could potentially be used for remote code execution. You should consider blocking open access to ports on EMR clusters from the Internet and restricting access only to specific IP addresses that require access to these ports.

For more information, see Remediating a Compromised EC2 Instance (p. 55).
Default severity: High

**Recon:IAMUser/TorIPCaller**

Finding description

*An API was invoked from a Tor exit node IP address*

This finding informs you that an API operation that can list or describe your AWS resources 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 be a reconnaissance attack: an anonymous user trying to gather information or gain access to your AWS resources for malicious purposes. For more information, see [Remediating Compromised AWS Credentials](p. 55).

Default severity: Medium

**Recon:IAMUser/MaliciousIPCaller.Custom**

Finding description

*An API was invoked from an IP address on a custom threat list.*

This finding informs you that an API operation that can list or describe your AWS resources 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. GuardDuty generates findings based on uploaded threat lists. This can be a reconnaissance attack: an anonymous user trying to gather information or gain access to your AWS resources for malicious purposes. For more information, see [Remediating Compromised AWS Credentials](p. 55).

Default severity: Medium

**Recon:IAMUser/MaliciousIPCaller**

Finding description

*An API was invoked from a known malicious IP address.*

This finding informs you that an API operation that can list or describe your AWS resources was invoked from an IP address that is included on a threat list. In GuardDuty, a threat list consists of known malicious IP addresses. GuardDuty generates findings based on the custom or internal threat lists. This can be a reconnaissance attack: an anonymous user trying to gather information or gain access to your AWS resources for malicious purposes. For more information, see [Remediating Compromised AWS Credentials](p. 55).

Default severity: Medium

**Recon:EC2/Portscan**

Finding description

**EC2 instance is performing outbound port scans to a remote host.**

This finding informs you that there is an EC2 instance in your AWS environment that 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 what services the machine is running.
and to identify its operating system. Your EC2 instance might be compromised. For more information, see Remediating a Compromised EC2 Instance (p. 55).

**Default severity: Medium**

**Recon:IAMUser/NetworkPermissions**

**Finding description**

A principal invoked an API commonly used to discover the network access permissions of existing security groups, ACLs, and routes in your AWS account.

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 invoking this API. Your credentials might be compromised. For more information, see Remediating Compromised AWS Credentials (p. 55).

This finding is triggered when network configuration settings in your AWS environment are probed under suspicious circumstances. For example, if a principal in your AWS environment with no prior history of doing so, invoked the DescribeSecurityGroups API. An attacker might use stolen credentials to perform this reconnaissance of network configuration settings before executing the next stage of their attack by changing network permissions or making use of existing openings in the network configuration.

**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 Amazon EC2 instance, the finding's severity is High.

**Recon:IAMUser/ResourcePermissions**

**Finding description**

A principal invoked an API commonly used to discover the permissions associated with various resources in your AWS account.

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 invoking this API. Your credentials might be compromised. For more information, see Remediating Compromised AWS Credentials (p. 55).

This finding is triggered when resource access permissions in your AWS account are probed under suspicious circumstances. For example, if a principal with no prior history of doing so, invoked the DescribeInstances API. An attacker might use stolen credentials to perform this reconnaissance of your AWS resources in order to find valuable information or determine the capabilities of the credentials they already have.

**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 Amazon EC2 instance, the finding's severity is High.

**Recon:IAMUser/UserPermissions**

**Finding description**

A principal invoked an API commonly used to discover the users, groups, policies and permissions in your AWS account.
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 invoking this API. Your credentials might be compromised. For more information, see Remediating Compromised AWS Credentials (p. 55).

This finding is triggered when user permissions in your AWS environment are probed under suspicious circumstances. For example, if a principal with no prior history of doing so, invoked the ListInstanceProfilesForRole API. An attacker might use stolen credentials to perform this reconnaissance of your IAM users and roles in order to determine the capabilities of the credentials they already have or to find more permissive credentials that are vulnerable to lateral movement.

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 Amazon EC2 instance, the finding’s severity is High.

ResourceConsumption Finding Types

This section covers the active ResourceConsumption threat purpose finding types. For information about important changes to the GuardDuty finding types, including newly added or retired finding types, see Document History for Amazon GuardDuty (p. 91).

Important
The default severity value of a finding type is subject to change based on various criteria when the finding is generated.

Topics
- ResourceConsumption:IAMUser/ComputeResources (p. 43)

ResourceConsumption:IAMUser/ComputeResources

Finding description

A principal invoked an API commonly used to launch compute resources like EC2 Instances.

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 invoking this API. Your credentials might be compromised. For more information, see Remediating Compromised AWS Credentials (p. 55).

This finding is triggered when EC2 instances in your AWS environment are launched under suspicious circumstances. For example, if a principal with no prior history of doing so, invoked the RunInstances API. 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.

Default severity: Medium

This finding’s default severity is Medium. However, if the API is invoked using temporary AWS credentials that are created on an Amazon EC2 instance, the finding’s severity is High.

Stealth Finding Types

This section covers the active Stealth threat purpose finding types. For information about important changes to the GuardDuty finding types, including newly added or retired finding types, see Document History for Amazon GuardDuty (p. 91).
Important
The default severity value of a finding type is subject to change based on various criteria when the finding is generated.

Topics
- Stealth:IAMUser/S3ServerAccessLoggingDisabled (p. 44)
- Stealth:IAMUser/PasswordPolicyChange (p. 44)
- Stealth:IAMUser/CloudTrailLoggingDisabled (p. 44)
- Stealth:IAMUser/LoggingConfigurationModified (p. 45)

Stealth:IAMUser/S3ServerAccessLoggingDisabled
Finding description
Amazon S3 server access logging was disabled for a bucket
This finding informs you that Amazon S3 server access logging is disabled for a bucket within your AWS environment. If disabled, no logs are created for any actions taken on the identified S3 bucket, or the objects in the bucket, unless Amazon S3 object level logging is enabled for this bucket. Disabling logging is a technique often used by unauthorized users in order to cover their tracks. This finding is triggered when Amazon S3 server access logging is disabled for a bucket. To learn more, see S3 Server Access Logging.

If you did not disable Amazon S3 server access logging for the identified bucket, it may indicate that your credentials are compromised. For more information, see Remediating Compromised AWS Credentials (p. 55).

Default severity: Low

Stealth:IAMUser/PasswordPolicyChange
Finding description
Account password policy was weakened.
Your AWS account password policy was weakened. 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. For more information, see Remediating Compromised AWS Credentials (p. 55).

Default severity: Low

Stealth:IAMUser/CloudTrailLoggingDisabled
Finding description
AWS CloudTrail trail was disabled.
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
Trojan Finding Types

This section covers the active Trojan threat purpose finding types. For information about important changes to the GuardDuty finding types, including newly added or retired finding types, see Document History for Amazon GuardDuty (p. 91).

Important
The default severity value of a finding type is subject to change based on various criteria when the finding is generated.

Topics
- Trojan:EC2/BlackholeTraffic (p. 45)
- Trojan:EC2/DropPoint (p. 46)
- Trojan:EC2/BlackholeTraffic!DNS (p. 46)
- Trojan:EC2/DriveBySourceTraffic!DNS (p. 46)
- Trojan:EC2/DropPoint!DNS (p. 46)
- Trojan:EC2/DGADomainRequest.B (p. 47)
- Trojan:EC2/DGADomainRequest.C!DNS (p. 47)
- Trojan:EC2/DNSDataExfiltration (p. 47)
- Trojan:EC2/PhishingDomainRequest!DNS (p. 48)

Trojan:EC2/BlackholeTraffic

Finding description

EC2 instance is attempting to communicate with an IP address of a remote host that is a known black hole.
This finding informs you that an 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 refer to 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. For more information, see Remediating a Compromised EC2 Instance (p. 55).

**Default severity: Medium**

**Trojan:EC2/DropPoint**

**Finding description**

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.

This finding informs you that an EC2 instance in your AWS environment is trying communicate with an IP address of a remote host that is known to hold credentials and other stolen data captured by malware. Your EC2 instance might be compromised. For more information, see Remediating a Compromised EC2 Instance (p. 55).

**Default severity: Medium**

**Trojan:EC2/BlackholeTraffic!DNS**

**Finding description**

EC2 instance is querying a domain name that is being redirected to a black hole IP address.

This finding informs you that an 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 refer to 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. For more information, see Remediating a Compromised EC2 Instance (p. 55).

**Default severity: Medium**

**Trojan:EC2/DriveBySourceTraffic!DNS**

**Finding description**

EC2 instance is querying a domain name of a remote host that is a known source of Drive-By download attacks.

This finding informs you that an 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 install of a virus, spyware, or malware. For more information, see Remediating a Compromised EC2 Instance (p. 55).

**Default severity: High**

**Trojan:EC2/DropPoint!DNS**

**Finding description**

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.
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. Your EC2 instance might be compromised. For more information, see Remediating a Compromised EC2 Instance (p. 55).

Default severity: High

Trojan:EC2/DGADomainRequest.B

Finding description

EC2 instance is querying algorithmically generated domains. Such domains are commonly used by malware and could be an indication of a compromised EC2 instance.

This finding informs you that there is an EC2 instance in your AWS environment that is trying to query domain generation algorithms (DGA) domains. Your EC2 instance might be compromised.

Note

This finding is based on analysis of domain names using advanced heuristics, and hence may identify new DGA domains that are not present in Threat Intelligence feeds.

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. C&C 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. For more information, see Remediating a Compromised EC2 Instance (p. 55).

Default severity: High

Trojan:EC2/DGADomainRequest.C!DNS

Finding description

EC2 instance is querying algorithmically generated domains. Such domains are commonly used by malware and could be an indication of a compromised EC2 instance.

This finding informs you that there is an EC2 instance in your AWS environment that is trying to query domain generation algorithms (DGA) domains. Your EC2 instance might be compromised.

Note

This finding is based on "known" DGA domains from GuardDuty’s threat intelligence feeds.

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. C&C 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. For more information, see Remediating a Compromised EC2 Instance (p. 55).

Default severity: High

Trojan:EC2/DNSDataExfiltration

Finding description

EC2 instance is exfiltrating data through DNS queries.
This finding informs you that there is an EC2 instance in your AWS environment with malware that uses DNS queries for outbound data transfers. The result is the exfiltration of data. Your EC2 instance might be compromised. 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. For more information, see Remediating a Compromised EC2 Instance (p. 55).

Default severity: High

Trojan:EC2/PhishingDomainRequest!DNS

Finding description

EC2 instance is querying domains involved in phishing attacks. Your EC2 instance might be compromised.

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 into providing sensitive data such as personally identifiable information, banking and credit card details, and passwords. Your EC2 instance is potentially trying to retrieve sensitive data stored on a phishing website. Or your EC2 instance is attempting to setup a phishing website. Your EC2 instance might be compromised. For more information, see Remediating a Compromised EC2 Instance (p. 55).

Default severity: High

Unauthorized Finding Types

This section covers the active Unauthorized threat purpose finding types. For information about important changes to the GuardDuty finding types, including newly added or retired finding types, see Document History for Amazon GuardDuty (p. 91).

Important

The default severity value of a finding type is subject to change based on various criteria when the finding is generated.

Topics

- UnauthorizedAccess:EC2/MetadataDNSRebind (p. 49)
- UnauthorizedAccess:IAMUser/TorIPCaller (p. 49)
- UnauthorizedAccess:IAMUser/MaliciousIPCaller:Custom (p. 49)
- UnauthorizedAccess:IAMUser/ConsoleLoginSuccess.B (p. 50)
- UnauthorizedAccess:IAMUser/MaliciousIPCaller (p. 50)
- UnauthorizedAccess:EC2/TorIPCaller (p. 50)
- UnauthorizedAccess:EC2/MaliciousIPCaller:Custom (p. 50)
- UnauthorizedAccess:EC2/SSHBruteForce (p. 51)
- UnauthorizedAccess:EC2/RDPBruteForce (p. 51)
- UnauthorizedAccess:IAMUser/InstanceCredentialExfiltration (p. 51)
- UnauthorizedAccess:IAMUser/ConsoleLogin (p. 52)
- UnauthorizedAccess:EC2/TorClient (p. 52)
- UnauthorizedAccess:EC2/TorRelay (p. 53)
UnauthorizedAccess:EC2/ MetadataDNSRebind

Finding description

An Amazon EC2 instance is performing DNS lookups that resolve to the instance metadata service.

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 which 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 a 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 a human user accesses the URL in a web browser running on the EC2 instance.

In response to this finding, you should evaluate whether there is a vulnerable application running on the EC2 instance, or a human user 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 it was due to a user browsing the identified domain, you should block the domain or prevent users from accessing it. If you determine this was related to either case above you should 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. Such customers can implement an archive filter to auto-archive all findings which have the type of UnauthorizedAccess:EC2/MetadataDNSRebind and the service.action.dnsRequestAction.domain field is same as the domain name they have mapped to the metadata IP address (169.254.169.254). To learn more, see CreateFilter.

Default severity: High

UnauthorizedAccess:IAMUser/TorIPCaller

Finding description

An API was invoked from a Tor exit node IP address.

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. For more information, see Remediating Compromised AWS Credentials (p. 55).

Default severity: Medium

UnauthorizedAccess:IAMUser/MaliciousIPCaller.Custom

Finding description

An API was invoked from an IP address on a custom threat list.

This finding informs you that an API operation (for example, an attempt to launch an EC2 instance, create a new IAM user, modify your AWS privileges, and so on) 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. GuardDuty generates findings based on uploaded threat lists. This can indicate unauthorized access to your AWS resources with the intent of hiding the attacker’s true identity. For more information, see Remediating Compromised AWS Credentials (p. 55).

Default severity: Medium

UnauthorizedAccess:IAMUser/ConsoleLoginSuccess.B

Finding description

Multiple worldwide successful console logins were observed.

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 pattern indicates potential unauthorized access to your AWS resources. For more information, see Remediating Compromised AWS Credentials (p. 55).

Note
This finding is only triggered by the activity of the following IAM identities: root, IAM users, and federated users. This finding is NOT triggered by the activity of an assumed role. For more information about IAM identities, see CloudTrail userIdentity Element.

Default severity: Medium

UnauthorizedAccess:IAMUser/MaliciousIPCaller

Finding description

An API was invoked from a known malicious IP address.

This finding informs you that an API operation (for example, an attempt to launch an EC2 instance, create a new IAM user, modify your AWS privileges, and so on) was invoked from a known malicious IP address. This can indicate unauthorized access to your AWS resources. For more information, see Remediating Compromised AWS Credentials (p. 55).

Default severity: Medium

UnauthorizedAccess:EC2/TorIPCaller

Finding description

EC2 instance is receiving inbound connections from a Tor exit node.

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. This can indicate unauthorized access to your AWS resources with the intent of hiding the attacker’s true identity. For more information, see Remediating a Compromised EC2 Instance (p. 55).

Default severity: Medium

UnauthorizedAccess:EC2/MaliciousIPCaller.Custom

Finding description

EC2 instance is communicating outbound with an IP address on a custom threat list.
Unauthorized Finding Types

This finding informs you that an EC2 instance in your AWS environment is communicating outbound 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. This can indicate unauthorized access to your AWS resources. For more information, see Remediating a Compromised EC2 Instance (p. 55).

Default severity: Medium

UnauthorizedAccess:EC2/SSHBruteForce

Finding description

EC2 instance has been involved in SSH brute force attacks.

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.

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.

Note
This finding is generated only through GuardDuty monitoring traffic on port 22. If your SSH services are configured to use other ports, this finding is not generated.

For more information, see Remediating a Compromised EC2 Instance (p. 55).

Default severity: Low

UnauthorizedAccess:EC2/RDPBruteForce

Finding description

EC2 instance has been involved in RDP brute force attacks.

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.

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.

For more information, see Remediating a Compromised EC2 Instance (p. 55).

Default severity: Low

UnauthorizedAccess:IAMUser/InstanceCredentialExfiltration

Finding description

Credentials that were created exclusively for an EC2 instance through an instance launch role are being used from an external IP address.

This finding informs you of attempts to run AWS API operations from a host outside of EC2, using temporary AWS credentials that were created on an EC2 instance in your AWS account. Your EC2 instance might be compromised, and the temporary credentials from this instance might have been exfiltrated to...
Unauthorized Finding Types

UnauthorizedAccess:IAMUser/InstanceCredentialExfiltration

This finding is generated when Amazon VPC networking is configured to route Internet traffic such that it egresses from an on-premise gateway rather than from a VPC Internet Gateway (IGW). Common configurations, such as using AWS Direct Connect, AWS Outposts, or VPC VPN connections, can result in traffic routed this way. To suppress this expected behavior, it’s recommended that you use the auto-archiving feature in GuardDuty and create a rule that consists of two filter criteria. The first criteria is “finding type”, which should be UnauthorizedAccess:IAMUser/InstanceCredentialExfiltration. The second filter criteria is either the IP address or the ASN of your on-premise internet gateway. For IP-based filters, use the “API caller IPv4 Address” criteria. For ASN based filters, use either the “API caller ASN name” or “API caller ASN ID”. GuardDuty still generates findings that match an auto-archiving filter rule. However, the rule causes these findings to go directly to the findings archive and not trigger an event for a CloudWatch Events rule or be sent to any other downstream integrations. To learn more, see Filtering Findings (p. 26).

Default severity: High

UnauthorizedAccess:IAMUser/ConsoleLogin

Finding description

An unusual console login by a principal in your AWS account was observed.

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. Your credentials might be compromised. For more information, see Remediating Compromised AWS Credentials (p. 55).

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).

Note
Unlike UnauthorizedAccess:IAMUser/ConsoleLoginSuccess.B, this finding can be triggered by any user type.

Default severity: Medium

UnauthorizedAccess:EC2/TorClient

Finding description

EC2 instance is making connections to a Tor Guard or an Authority node.

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. For more information, see Remediating a Compromised EC2 Instance (p. 55).

Default severity: High

UnauthorizedAccess:EC2/TorRelay

Finding description

EC2 instance is making connections to a Tor network as a Tor relay.

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 relays increase anonymity of the communication by forwarding the client’s possibly illicit traffic from one Tor relay to another. If this activity is unexpected, your EC2 instance might be compromised. For more information, see Remediating a Compromised EC2 Instance (p. 55).

Default severity: High

For information about important changes to the GuardDuty finding types, including newly added or retired finding types, see Document History for Amazon GuardDuty (p. 91).

Retired Finding Types

Important
For information about important changes to the GuardDuty finding types, including newly added or retired finding types, see Document History for Amazon GuardDuty (p. 91).

In the current release of GuardDuty, the following finding types are retired (no longer generated). You CANNOT reactivate retired GuardDuty findings types.

Topics
- Backdoor:EC2/XORDDOS (p. 53)
- Behavior:IAMUser/InstanceLaunchUnusual (p. 54)
- CryptoCurrency:EC2/BitcoinTool.A (p. 54)
- UnauthorizedAccess:IAMUser/UnusualASNCaller (p. 54)

Backdoor:EC2/XORDDOS

Default severity: High

Finding description

An EC2 instance is attempting to communicate with an IP address that is associated with XorDDos malware.

This finding informs you that an EC2 instance in your AWS environment is attempting to communicate with an IP address that is associated with XorDDos 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 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. For more information, see Remediating a Compromised EC2 Instance (p. 55).

**Behavior:IAMUser/InstanceLaunchUnusual**

Finding description

An IAM user launched an EC2 instance of an unusual type.

This finding informs you that a specific IAM user in your AWS environment is exhibiting behavior that is different from the established baseline. This IAM user has no prior history of launching an EC2 instance of this type. Your IAM user credentials might be compromised. For more information, see Remediating Compromised AWS Credentials (p. 55)

**CryptoCurrency:EC2/BitcoinTool.A**

Finding description

EC2 instance is communicating with Bitcoin mining pools.

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. For more information, see Remediating a Compromised EC2 Instance (p. 55).

**UnauthorizedAccess:IAMUser/UnusualASNCaller**

Finding description

An API was invoked from an IP address of an unusual network.

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. For more information, see Remediating Compromised AWS Credentials (p. 55).

**Important**

The UnauthorizedAccess:IAMUser/UnusualASNCaller finding type has been retired. You will now be notified about activity invoked from unusual networks via one of the following active GuardDuty finding types, based on the category of the API that was invoked from an unusual network:

- Persistence:IAMUser/NetworkPermissions
- Persistence:IAMUser/ResourcePermissions
- Persistence:IAMUser/UserPermissions
- Recon:IAMUser/NetworkPermissions
- Recon:IAMUser/ResourcePermissions
- Recon:IAMUser/UserPermissions
- ResourceConsumption:IAMUser/ComputeResources
Remediating Security Issues Discovered by GuardDuty

Amazon GuardDuty generates findings (p. 23) that indicate potential security issues. In this release of GuardDuty, the potential security issues indicate either a compromised EC2 instance or a set of compromised credentials in your AWS environment. The following sections describe the recommended remediation steps for either scenario.

Topics
- Remediating a Compromised EC2 Instance (p. 55)
- Remediating Compromised AWS Credentials (p. 55)

Remediating a Compromised EC2 Instance

Follow these recommended steps to remediate a compromised EC2 instance in your AWS environment:

- Investigate the potentially compromised instance for malware and remove any discovered malware. You can also refer to the AWS Marketplace for partner products that might help to identify and remove malware.
- If you are unable to identify and stop unauthorized activity on your 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:
  - Amazon EC2 Security Groups for Linux Instances and Amazon EC2 Security Groups for Windows Instances.
  - Tips for securing your EC2 instances (Linux).
  - AWS Security Best Practices
  - Browse for further assistance on the AWS developer forums: https://forums.aws.amazon.com/index.jspa
  - If you are a Premium Support package subscriber, you can submit a technical support request.

Remediating Compromised AWS Credentials

Follow these recommended steps to remediate compromised credentials in your AWS environment:

- Identify the owner of the credentials.

If a GuardDuty finding informs you of a potential compromise to AWS credentials, you can locate the affected IAM user by their access keys or user name.

Note
Users need their own access keys to make programmatic calls to AWS from the AWS Command Line Interface (AWS CLI), Tools for Windows PowerShell, the AWS SDKs, or direct HTTP calls using the APIs for individual AWS services. To fill this need, you can create, modify, view, or rotate access keys (access key IDs and secret access keys) for IAM users. For more information, see Managing Access Keys for IAM Users.
To find the access key ID or user name that belongs to a potentially compromised IAM user, open the console and view the details pane of the finding that you're analyzing. For more information, see Locating and Analyzing GuardDuty Findings (p. 23). After you have the access key ID or user name, open the IAM console, choose the Users tab, and locate the affected user by typing the access key ID or user name in the Find users by username or access key search field.

- **Determine whether the credentials were used by the IAM user legitimately.**

  Contact the IAM user that you've located, and verify whether the user legitimately used the access key and user name that is identified in the GuardDuty finding. 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 you confirm that the activity is a legitimate use of the AWS credentials, you can ignore the GuardDuty finding. If not, this activity is likely the result of a compromise to that particular access key, the IAM user's user ID and password, or possibly the entire AWS account. You can then use the information in the My AWS account may be compromised article to remediate the issue.
Exporting Findings

GuardDuty supports exporting active findings to CloudWatch Events and, optionally, to an Amazon S3 bucket. 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 updated Active findings are exported to CloudWatch Events and your S3 bucket (if configured). The frequency that you select applies to exporting to both CloudWatch Events and your S3 bucket, but only for updated findings.

When you configure options for exporting findings, the settings apply only to the account in the current AWS Region. Only findings that are generated in the current account and Region are exported. If you enable findings export in a GuardDuty master account, all findings from associated member accounts that are generated in the current Region are also exported to the same location that is configured for the master account.

Because GuardDuty is enabled per Region, to export findings from all Regions, you need to configure export options for each Region in which you’re using GuardDuty. You can export findings from all Regions into one bucket located in one Region. You can also choose to export findings to a bucket in a different account.

**Important**
Archived findings, including new instances of auto-archived findings, aren’t exported. If you unarchive a finding, its status is updated to Active, and then it’s exported.

To learn more, see Monitoring GuardDuty Findings with Amazon CloudWatch Events (p. 80).

Permissions Required to Configure Findings Export

When you configure options for exporting findings, you select a bucket to store the findings, and a KMS key to use for data encryption. In addition to permissions to GuardDuty actions, you must also have permissions to the following actions to successfully configure options for exporting findings.

- kms:ListAliases
- s3:CreateBucket
- s3:GetBucketLocation
- s3:ListAllMyBuckets
- s3:PutBucketAcl
- s3:PutBucketPublicAccessBlock
- s3:PutBucketPolicy
- s3:PutObject

Setting the Frequency for Exporting 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. Sign in to the AWS Management Console and open the GuardDuty console at https://console.aws.amazon.com/guardduty/.
2. Choose Settings.
3. In the Findings export options section, select Frequency for updated findings. This sets the frequency for exporting updated Active findings to both CloudWatch Events and Amazon S3. You can choose from the following:
   - Update CWE and S3 every 15 minutes
   - Update CWE and S3 every 1 hour
   - Update CWE and S3 every 6 hours (default)
4. Choose Save.

Configuring Findings Export to an Amazon S3 Bucket

Configure settings for exporting Active findings to an Amazon S3 bucket from the Settings page in the GuardDuty console. When you configure findings export, you can choose an existing S3 bucket, or have GuardDuty create a new bucket to store exported findings. If you choose to use a new bucket, GuardDuty applies all necessary permissions to the created bucket. If you use an existing bucket, you must update the bucket policy to allow GuardDuty to put findings into the bucket.

Important
The bucket and KMS key that you use must be in the same Region.

Changing the Key Policy

GuardDuty encrypts the findings data in the bucket by using an AWS KMS key. To successfully configure findings export, you must first give GuardDuty permission to use the key that you choose when you configure findings export. You grant the permissions by changing the key policy for the key you use.

If you plan to use a new key for GuardDuty findings, create a key before proceeding. If the key isn’t in your account, you need to log in to the account that owns the key to apply the key policy. You also need the key ARN for a key from another account when you configure export settings.

To modify the key policy to allow GuardDuty to use the key

1. Determine which key you want to use, either an existing or a new customer master key (CMK). The key must be in the same Region as the bucket.
2. Open the AWS KMS console at https://console.aws.amazon.com/kms.
3. To change the AWS Region, use the Region selector in the upper-right corner of the page.
4. Choose the key that you plan to use for encrypting exported findings.
5. Choose Key policy, and then choose Edit.
   Note
   If Switch to policy view is displayed, choose that to display the key policy, and then choose Edit.
6. Add the following policy statement to the policy. The statement allows GuardDuty to use only the key that you changed the policy for.
Changing the Key Policy

```json
{
  "Sid": "Allow GuardDuty to use the key",
  "Effect": "Allow",
  "Principal": {
    "Service": "guardduty.amazonaws.com"
  },
  "Action": "kms:GenerateDataKey",
  "Resource": "*"
}
```

**Note**

If you're using GuardDuty in an manually-enabled Region, replace the value for the "Service" with the service endpoint for the 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".

7. Choose Save.

When you add the statement to the policy, make sure that the syntax is valid. Policies are displayed in JavaScript Object Notation (JSON) format. When you add a new section to the policy, you need to include a comma either before or after the added section. The position of the comma depends on how you add the new statement. If you add the statement as the last statement in the policy, add a comma after the closing bracket for the preceding section. If you add it as the first statement, or add it between two existing statements, add a comma after the closing bracket. The following examples show how to add the policy statement to a default key policy.

The following example shows a default key policy with no additional permissions granted:

```json
{
  "Id": "key-consolepolicy",
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "Enable IAM User Permissions",
      "Effect": "Allow",
      "Principal": {
        "AWS": "arn:aws:iam::111122223333:root"
      },
      "Action": "kms:*",
      "Resource": "*"
    }
  ]
}
```

The following example shows how to add the policy statement as the first statement:

```json
{
  "Id": "key-consolepolicy",
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "Allow GuardDuty to use the key",
      "Effect": "Allow",
      "Principal": {"Service": "guardduty.amazonaws.com"},
      "Action": "kms:GenerateDataKey",
      "Resource": "*"
    },
    {// Add a comma after this bracket
    {
      "Sid": "Enable IAM User Permissions",
    
```
The following example shows how to add the policy statement as the last statement:

```
{
  "Id": "key-consolepolicy",
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "Enable IAM User Permissions",
      "Effect": "Allow",
      "Principal": {"AWS": "arn:aws:iam::111122223333:root"},
      "Action": "kms:*",
      "Resource": "*"
    },
    {
      "Sid": "Allow GuardDuty to use the key",
      "Effect": "Allow",
      "Principal": {"Service": "guardduty.amazonaws.com"},
      "Action": "kms:GenerateDataKey",
      "Resource": "*"
    }
  ]
}
```

Exporting Findings to a New Bucket

When you choose to create a new bucket to configure findings export, GuardDuty adds a bucket policy to the bucket that grants GuardDuty permission to put findings into the bucket.

**To configure findings export using a new bucket**

2. Choose **Settings**.
3. Choose **New bucket** to create a new bucket to store exported findings.
   
   In the **Name the bucket** field, enter a name for the bucket. The name must be unique across all S3 buckets. Bucket names must start with a lowercase letter or a number.
4. Optional. Under **Log file prefix**, enter the path prefix to use in the path to the location in the bucket where findings are stored. When you enter a value, the example path below the field is updated to reflect the path to exported findings in the bucket.
5. Under **KMS encryption**, do one of the following:
   - Select **Choose key from your account**.
     
     Then choose the key alias of the key that you changed the policy for from the **Key alias** list.
   - Select **Choose key from another account**.
     
     Then enter the full ARN to the key that you changed the policy for.
   
   The key that you choose must be in the same Region as the bucket. To learn how to find the key ARN, see [Finding the Key ID and ARN](#).
6. Choose **Save**.
Exporting Findings to an Existing Bucket

When you create a new bucket to export findings to, GuardDuty modifies the bucket policy to grant GuardDuty the necessary permissions to put findings in the bucket. If you choose to use an existing bucket, you must update the bucket policy to grant GuardDuty those permissions to successfully configure findings export settings. To learn more, see How Do I Add an S3 Bucket Policy?

Granting GuardDuty Permissions to Export Findings to Your Bucket

To add a bucket policy

Open the Amazon S3 console at https://console.aws.amazon.com/s3/.

1. Choose the bucket you plan to use for exported findings.
2. Choose Permissions, and then choose Bucket Policy.
3. Copy the example policy and paste it into the Bucket policy editor.
4. Replace the placeholder values in the example policy with the values appropriate for your environment.

Replace myBucketName with the name of the bucket that you're adding the bucket policy for. Replace region with the Region that the KMS key is in. Replace 111122223333 with your AWS account number, and replace KMSKeyId with the key ID of the key that you chose for encryption.

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

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "Allow GuardDuty to use the getBucketLocation operation",
      "Effect": "Allow",
      "Principal": {
        "Service": "guardduty.amazonaws.com"
      },
      "Action": "s3:GetBucketLocation",
      "Resource": "arn:aws:s3:::myBucketName"
    },
    {
      "Sid": "Allow GuardDuty to upload objects to the bucket",
      "Effect": "Allow",
      "Principal": {
        "Service": "guardduty.amazonaws.com"
      },
      "Action": "s3:PutObject",
      "Resource": "arn:aws:s3:::myBucketName/[optional prefix]/*"
    },
    {
      "Sid": "Deny unencrypted object uploads. This is optional",
      "Effect": "Deny",
      "Principal": {
        "Service": "guardduty.amazonaws.com"
      },
      "Action": "s3:PutObject",
      "Resource": "arn:aws:s3:::myBucketName/[optional prefix]/**",
      "Condition": {
```
"StringNotEquals": {
    "s3:x-amz-server-side-encryption": "aws:kms"
  },
},
{
  "Sid": "Deny incorrect encryption header. This is optional",
  "Effect": "Deny",
  "Principal": {
    "Service": "guardduty.amazonaws.com"
  },
  "Action": "s3:PutObject",
  "Resource": "arn:aws:s3:::myBucketName/[optional prefix]/*",
  "Condition": {
    "StringNotEquals": {
      "s3:x-amz-server-side-encryption-aws-kms-key-id": "arn:aws:kms:region:111122223333:key/KMSKeyId"
    }
  }
},
{
  "Sid": "Deny non-HTTPS access",
  "Effect": "Deny",
  "Principal": "*",
  "Action": "s3:*",
  "Resource": "arn:aws:s3:::myBucketName/*",
  "Condition": {
    "Bool": {
      "aws:SecureTransport": "false"
    }
  }
}
]

**Note**
If you're using GuardDuty in an manually-enabled Region, replace the value for the service with the service endpoint for the Region. For example, if you're using GuardDuty in the Middle East (Bahrain) (me-south-1) Region, replace "Service": "guardduty.amazonaws.com" with "guardduty.me-south-1.amazonaws.com".

**To configure findings export using an existing bucket**

1. Identify the bucket that you plan to export findings to.
2. Open the GuardDuty console at https://console.aws.amazon.com/guardduty/.
3. Choose **Settings**.
4. Under **S3 bucket**, choose **Configure now**.
5. Do one of the following:
   - Choose **Existing bucket in your account** to choose a bucket in your account.
     Under **Choose a bucket**, choose the bucket from your account to use to store exported findings.
   - Choose **Existing bucket in another account**.
     In the **Bucket ARN field**, enter the ARN for the bucket from another account to use. The ARN can be for any bucket, including buckets in the current account.
6. Optional. Under **Log file prefix**, enter the path prefix to use in the path to the location in the bucket where findings are stored. When you enter a value, the example path below the field is updated to reflect the path to exported findings in the bucket.
7. Under **KMS encryption**, do one of the following:
- Select **Choose key from your account**.
  Then choose the key alias of the key that you changed the policy for from the **Key alias** list.
- Select **Choose key from another account**.
  Then enter the full ARN to the key that you changed the policy for.

The key you choose must be in the same Region as the bucket. To learn how to find the key ARN, see **Finding the Key ID and ARN**.

8. Choose **Save**.

## Export Access Error

After you configure finding export options, if GuardDuty is unable to export findings, an error message is displayed on the **Settings** page. This can happen when GuardDuty can no longer access the target resource, such as when the S3 bucket is deleted or the permissions to the bucket are changed. This can also happen when the KMS key used to encrypt data in the bucket becomes inaccessible.

When exporting fails, GuardDuty sends a notification to the email associated with the account to let you know about the issue. If you don't resolve the issue, GuardDuty disables finding export in the account. You can update the configuration to restart finding export at any time.

If you receive this error, review the information in this topic about how to enable and configure findings export. For example, review the key policy and confirm that the correct policy is applied to the KMS key that you chose for encryption.
Working with Trusted IP Lists and Threat Lists

Amazon GuardDuty monitors the security of your AWS environment by analyzing and processing VPC Flow Logs, AWS CloudTrail event logs, and DNS logs. You can customize this monitoring scope by configuring GuardDuty to also use your own trusted IP lists and threat lists. The IP lists described below will apply to all VPC Flow Log and CloudTrail findings, but do not apply to DNS findings.

Trusted IP lists consist of IP addresses that you have whitelisted for secure communication with your AWS infrastructure and applications. GuardDuty does not generate VPC Flow Log or CloudTrail findings for IP addresses on trusted IP lists. At any given time, you can have only one uploaded trusted IP list per AWS account per Region.

Threat lists consist of known malicious IP addresses. GuardDuty generates findings based on threat lists. At any given time, you can have up to six uploaded threat lists per AWS account per Region.

Users from master GuardDuty accounts can upload and manage trusted IP lists and threat lists. Users from member GuardDuty accounts CANNOT upload and manage lists. Trusted IP lists and threat lists that are uploaded by the master account are imposed on GuardDuty functionality in its member accounts. In other words, in member accounts GuardDuty generates findings based on activity that involves known malicious IP addresses from the master's threat lists and does not generate findings based on activity that involves IP addresses from the master's trusted IP lists. For more information, see Managing Multiple Accounts in Amazon GuardDuty (p. 68).

**Important**

GuardDuty uses the same AWS_ServiceRoleForAmazonGuardDuty service-linked role (p. 8) that is automatically assigned to it when you enable GuardDuty for the permissions required to evaluate your trusted IP lists and threat lists.

Note the following when creating trusted IP lists and threat lists that you plan to upload with GuardDuty:

- In your trusted IP lists and threat lists, IP addresses and CIDR ranges must appear one per line.

The following is a sample list in Plaintext format:

```
54.20.175.217
205.0.0.0/8
```

- Trusted IP list and threat lists apply only to traffic destined for publicly routable IP addresses.
- GuardDuty doesn't generate findings based on activity that involves domain names that are included in your threat lists. GuardDuty only generates findings based on activity that involves IP addresses and CIDR ranges in your threat lists.
- The maximum size of the file that hosts your trusted IP list or threat list is 35MB.
- You can include a maximum of 2000 IP addresses and CIDR ranges in a single trusted IP list.
- You can include a maximum of 250,000 IP addresses and CIDR ranges in a single threat list.

**Topics**

- Permissions Required to Upload Trusted IP Lists and Threat Lists (p. 65)
- To Upload Trusted IP Lists and Threat Lists (p. 65)
- To Activate or Deactivate Trusted IP Lists and Threat Lists (p. 66)
Permissions Required to Upload Trusted IP Lists and Threat Lists

Various IAM identity require proper permissions to work with trusted IP lists and threat lists in GuardDuty. An identity with the AmazonGuardDutyFullAccess managed policy attached 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 uploading, activating, deleting, and updating the location of the lists), make sure that the following actions are present in the permissions policy attached to an IAM user, group, or role:

```json
{
  "Effect": "Allow",
  "Action": ["iam:PutRolePolicy", "iam:DeleteRolePolicy"],
  "Resource": "arn:aws:iam::123456789123:role/aws-service-role/guardduty.amazonaws.com/AWSServiceRoleForAmazonGuardDuty"
}
```

Important
These actions are not included in the AmazonGuardDutyFullAccess managed policy.

To Upload Trusted IP Lists and Threat Lists

The following procedure describes how you can upload trusted IP lists and threat lists using the GuardDuty console.

**To upload trusted IP lists and threat lists (console)**

2. In the navigation pane, under Settings, choose Lists.
3. On the List management page, choose Add a trusted IP list or Add a threat list.
4. In the dialog box, do the following:
   - For **List name**, type a name for the list.
   - For **Location**, specify the location of the list - this is the S3 bucket where you store your trusted IP list or threat list and the file that contains your list.

   **Note**
   You can specify the location URL in the following formats:
   - 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://mybucket/file.txt
To Activate or Deactivate Trusted IP Lists and Threat Lists

The following procedures describe how you can activate or deactivate trusted IP lists and threat lists in GuardDuty once they are uploaded. GuardDuty includes the uploaded lists in its monitoring of your AWS environment only if they are active.

To activate trusted IP lists and threat lists (console)
2. In the navigation pane, under Settings, choose Lists.
3. On the List management page, locate the trusted IP set or a threat list that you want to activate, and then choose the radio button under the Active column.

To deactivate trusted IP lists and threat lists (API or CLI)
- Currently in GuardDuty, deactivating trusted IP lists and threat lists through the console is not supported.

You can deactivate your trusted IP lists or threat lists by running the UpdateThreatIntelSet and UpdateIPSet operations, or the update-ip-set and update-threat-intel-set CLI commands.

For example, you can run the following command:

```
aws guardduty update-ip-set --detector-id <detector-id> --ip-set-id <ip-set-id> --no-activate
```

Make sure to replace <detector-id> and <ip-set-id> with a valid detector ID and trusted IP list ID.

To Update Trusted IP Lists and Threat Lists

If you make changes to a trusted IP list or a threat list that is already uploaded and activated in GuardDuty (for example, rename the list or add more IP addresses to it), you must update this list in GuardDuty and reactivate it in order for GuardDuty to use the latest version of the list in its security monitoring scope. To update a safe or threat list, you can either use the procedure below or run the UpdateThreatIntelSet and UpdateIPSet operations of the GuardDuty API.

To update trusted IP lists and threat lists (console)
2. In the navigation pane, under Settings, choose Lists.
3. On the List management page, locate the trusted IP set or a threat list that you want to update, and then choose the pencil icon under the Active column.
4. In the Update list pop up window, verify all specified list information, choose I agree, and then choose Update list.
5. On the **List management** page, locate the trusted IP set or a threat list that you want to activate again, and then choose the radio button under the **Active** column.

To learn how to import threat intel feeds programatically, see the *How to automate the import of third-party threat intelligence feeds* blog article.
Managing Multiple Accounts in Amazon GuardDuty

To manage multiple accounts in Amazon GuardDuty, you must choose a single AWS account to be the master account for GuardDuty. You can then associate other AWS accounts with the master account as member accounts. There are two ways to associate accounts with a GuardDuty master account: either through an AWS Organizations organization that both accounts are members of, or by sending an invitation through GuardDuty.

GuardDuty recommends using the AWS Organizations method. For more information about setting up an organization, see Creating an organization in the AWS Organizations User Guide.

Managing Multiple Accounts with Organizations

If the account that you want to specify as the GuardDuty master account is part of an organization in AWS Organizations, then you can specify that account as the organization’s delegated administrator for GuardDuty. The account that is registered as the delegated administrator automatically becomes the GuardDuty master account.

You can use the master account to enable GuardDuty for any account in the organization and then add that account as a member account.

If you already have a GuardDuty master account with associated member accounts by invitation, you can register that account as the GuardDuty delegated administrator for the organization. When you do, all currently associated member accounts remain members, allowing you to take full advantage of the added functionality of managing your GuardDuty accounts with AWS Organizations.

For more information about supporting multiple accounts in GuardDuty through an organization see Managing GuardDuty Accounts with AWS Organizations (p. 69).

Managing Multiple Accounts by Invitation

If the accounts you want to associate are not part of an AWS Organizations organization, you can specify a master account in GuardDuty and then use the master account to invite other AWS accounts to become member accounts. When the invited account accepts the invitation, that account becomes a GuardDuty member account associated with the master account.

For more information about supporting multiple accounts by Invitation in GuardDuty see Managing GuardDuty Accounts by Invitation (p. 73).

Understanding the Relationship between GuardDuty Master and Member Accounts

When you use GuardDuty in a multiple-account environment, the master account can manage certain aspects of GuardDuty on behalf of the member accounts. The primary functions the master account can perform are the following:

- Add and remove associated member accounts. The process by which this is done differs based on whether the accounts are associated through organizations or by invitation.
• Manage the status of GuardDuty within associated member accounts, including enabling and suspending GuardDuty.

  **Note**  
  Master accounts managed with AWS Organizations automatically enable GuardDuty in accounts added as members.

• Customize findings within the GuardDuty network through the creation and management of suppression rules, trusted IP lists, and threat lists. Member accounts lose access to these features in a multiple-account environment.

The following table details the relationship between GuardDuty master and member accounts.

Account designations listed as *Self* can take the listed action only in their own accounts. A designation of *Any* indicates that account can perform the described action for any associated account, and *All* denotes actions that are applied to all associated accounts when taken by the designated account.

<table>
<thead>
<tr>
<th>Action</th>
<th>Designation</th>
</tr>
</thead>
<tbody>
<tr>
<td>View accounts from your AWS Organizations organization</td>
<td>Any</td>
</tr>
<tr>
<td>Enable GuardDuty</td>
<td>Any</td>
</tr>
<tr>
<td>View GuardDuty findings</td>
<td>Any</td>
</tr>
<tr>
<td>Archive findings</td>
<td>Any</td>
</tr>
<tr>
<td>Apply suppression rules</td>
<td>All</td>
</tr>
<tr>
<td>Generate sample findings</td>
<td>Self</td>
</tr>
<tr>
<td>Create trusted IP or threat lists</td>
<td>All</td>
</tr>
<tr>
<td>Update trusted IP or threat lists</td>
<td>All</td>
</tr>
<tr>
<td>Delete trusted IP or threat lists</td>
<td>All</td>
</tr>
<tr>
<td>Set CloudWatch Events notification frequency</td>
<td>All</td>
</tr>
<tr>
<td>Set Amazon S3 location for exporting findings</td>
<td>All</td>
</tr>
<tr>
<td>Suspend GuardDuty</td>
<td>Any*</td>
</tr>
</tbody>
</table>

* Indicates this action must be taken for all associated accounts before being taken in the designated account.

### Managing GuardDuty Accounts with AWS Organizations

When you use GuardDuty with an AWS Organizations organization, you designate an account to be the GuardDuty delegated administrator for the organization. Only the organization master account can designate GuardDuty delegated administrators.
Designate a Delegated Administrator and add Member Accounts (Console)

Note
The Organization master can be the delegated administrator, but this is not recommended based on AWS Security best practices following the principle of least privilege.

An account that is designated as a delegated administrator becomes a GuardDuty master account, has GuardDuty automatically enabled in the designated Region, and is granted permission to enable and manage GuardDuty for all accounts in the organization within that Region. The other accounts in the organization can be viewed and added as GuardDuty member accounts associated with the master account.

If you have already set up a GuardDuty master with associated member accounts by invitation, and the member accounts are part of the same organization, their Type changes from by Invitation to via Organizations when you set a GuardDuty delegated administrator for your organization. If the organization’s GuardDuty master previously added members by invitation that are not part of the same organization, their Type is by Invitation. In both cases, these previously added accounts are member accounts to the organization’s GuardDuty delegated administrator.

You can continue to add accounts as members even if they are outside of your organization. To learn more, see Designating Master and Member Accounts Through Invitation (Console) (p. 74) and Designating GuardDuty Master and Member Accounts Through Invitation (API) (p. 75).

Note
There is a limit of 5000 member accounts per GuardDuty master account. However, there could be more than 5000 accounts in your organization. The number of All accounts in your organization is displayed on the Accounts page of the GuardDuty console. If you exceed 5000 member accounts you will receive a notification through CloudWatch, Personal Health Dashboard, and in an email to the master account.

Use the following procedures to register a GuardDuty delegated administrator for your organization, add existing organization accounts as members, and automate the addition of new organization accounts as GuardDuty member accounts.

Designate a Delegated Administrator and add Member Accounts (Console)

Follow these steps to designate a GuardDuty delegated administrator and add member accounts using the GuardDuty console.

Important
You need to repeat these steps in each Region where you are using GuardDuty if you want to monitor all Regions for organization member accounts associated with the delegated administrator.

Step 1 - Register a GuardDuty delegated administrator for your organization

1. Log in to the AWS Management Console using the master account for your AWS Organizations organization.
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 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.
Designate a Delegated Administrator and add Member Accounts (Console)

3. Enter the twelve-digit AWS account ID of the account that you want to designate as the GuardDuty delegated administrator for the organization.

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

5. (Recommended) Repeat the previous steps in each AWS Region.
   
   Note
   
   It is recommended that you register the same delegated administrator in each AWS Region.

After you designate the delegated administrator, you only need to use the master account organization to change or remove the delegated administrator account.

   Note
   
   If you remove the delegated administrator, all associated member accounts are removed as GuardDuty members, but GuardDuty is not disabled in those accounts.

Step 2 - Add existing organization accounts as members

When you add an account as a member, GuardDuty is automatically enabled in that account in the current Region.

You must add your organization members in each Region to enable GuardDuty for those Regions.


2. In the navigation panel, choose Settings, and then choose Accounts.

   The accounts table displays all of the accounts in the organization. The Type for these accounts is via organizations. The status of accounts that are not member accounts associated with the organization's GuardDuty delegated administrator is Not a member.

3. Select the account or accounts that you want to add as members by checking the box next to the account ID.

   Note
   
   You can enable GuardDuty in the current Region for all organization accounts by selecting enable in the banner at the top of the page. This action also triggers the Auto-Enable feature that will enable GuardDuty in any future accounts added to your organization. Alternately, you can use the filter field to filter by Relationship status: Not a member, and then select every account that does not have GuardDuty enabled in the current Region.

4. Choose Actions then choose Add member.

5. Confirm that you want to add as members the number of accounts selected. The Status for the invited accounts will change to Enabled.

6. (Recommended) Repeat these steps in each AWS Region to ensure that your delegated administrator can manage findings for member accounts in all Regions.

Step 3 - Automate the addition of new organization accounts as members

1. Log in to the https://console.aws.amazon.com/guardduty/ console using the delegated administrator account.

2. In the navigation pane, under Settings, choose Accounts.

3. Turn on Auto-enable.

4. Confirm your selection.
5. **(Recommended) Repeat these steps in each AWS Region to ensure that GuardDuty is automatically enabled on any new account, in every Region.**

When this setting is enabled, all new accounts that are created in, or added to, the organization are added as a member accounts of the organization's GuardDuty delegated administrator and GuardDuty is enabled in that Region. When the number of member accounts reaches the limit of 5000, the Auto-enable feature is automatically turned off. If an account is removed and the total number of members decreases to fewer than 5000, the Auto-enable feature turns back on.

**Designate a Delegated Administrator and add Member Accounts (API)**

You can designate an Organizations delegated administrator and GuardDuty member accounts through the API operations. Complete the following procedures to add a delegated administrator and member accounts in GuardDuty with organizations.

**Important**

You need to repeat these steps in each Region where you are using GuardDuty if you want to monitor all Regions for organization member accounts associated with the delegated administrator.

1. **Run the enableOrganizationAdminAccount API operation using the credentials of the AWS account of the Organizations master.**

   You must specify the account ID of the account you want to make a GuardDuty delegated administrator.

   You can also use the AWS Command Line to do this by running the following CLI command. Make sure to use a valid account ID.

   ```
   aws guardduty enable-organization-admin-account --admin-account-id 11111111111
   ```

   **Note**

   This command sets the delegated administrator for you current Region only. If GuardDuty is not already enabled for that account in the current Region, it will be automatically enabled.

   To set the delegated administrator for other Regions, you must specify the Regional endpoint for the Region you want your delegated administrator to manage. For more information, see GuardDuty endpoints and quotas. The following example demonstrates how to set your endpoint to US West (Oregon).

   ```
   aws guardduty enable-organization-admin-account --admin-account-id 11111111111 --endpoint guardduty.us-west-2.amazonaws.com
   ```

2. **Run the CreateMembers API operation using the credentials of the AWS account you designated as the delegated administrator for GuardDuty in the previous step.**

   You must specify the detector ID of the delegated administrator AWS account and the account details, including 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 do this using AWS Command Line Tools by running the following CLI command. Make sure to use your own valid detector ID, account ID, and email.
Managing Accounts by Invitation

```
aws guardduty create-members --detector-id 12abc34d567e8fa901bc2d34e56789f0 --account-details AccountId=123456789012,Email=guarddutymember@amazon.com
```

You can view a list of all organization members using the `ListAccounts` API operation or by running the following CLI command.

```
aws organizations list-accounts
```

**Note**
The detector ID is Regional. To enable GuardDuty for organization members in other Regions, you must supply the unique detector ID of the delegated administrator for each Region.

3. Run the `updateOrganizationConfiguration` API operation using the credentials of the GuardDuty delegated administrator account to automatically enable GuardDuty in that Region for new member accounts.

You must specify the detector ID of the delegated administrator AWS account.

**Note**
The detector ID is Regional, to automatically enable GuardDuty for new members in each Region, you must supply the unique detector ID of the delegated administrator for each Region.

You can also do this using AWS Command Line Tools by running the following CLI command. Make sure to use your own valid detector ID.

```
aws guardduty update-organization-configuration --detector-id 12abc34d567e8fa901bc2d34e56789f0 --auto-enable
```

You can confirm that you have turned on the auto enable GuardDuty feature in a Region by running the `describeOrganizationConfiguration` API operation or by running the following CLI command using the detector ID of the delegated administrator in the desired Region.

```
aws guardduty describe-organization-configuration --detector-id 12abc34d567e8fa901bc2d34e56789f0
```

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 master account when another account accepts your invitation to become a member account.

If your account is not a master 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 master and member account at the same time.

Accounts associated by invitation have the same overall master-to-member relationship as accounts associated by AWS Organizations, as described in Understanding the Relationship between GuardDuty Master and Member Accounts (p. 68). However, invitation master 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.

### Designating Master and Member Accounts Through Invitation (Console)

Use the following procedures to add an account, invite an account, or accept an invitation from another account.

#### Step 1 - Add an account

2. In the navigation pane, choose **Accounts**.
3. Choose **Add accounts by invitation** in the top panel.
4. On the **Add member accounts** page, under **Enter accounts**, enter the AWS account ID and email address of the account that you want to add. Then choose **Add**.

   **Important**
   The email address that you specify in this step MUST be identical to the email address associated with the AWS account that you want to add as GuardDuty member account.

   You can add more accounts, one at a time, by specifying their IDs and email addresses. You can also choose **Upload list (.csv)** to bulk add accounts. This can be useful if you want to invite some of these accounts to enable GuardDuty right away but want to delay for others.

   **Important**
   The first line of your CSV file must contain the following header, as shown in the following example: **Account ID,Email**. Each subsequent line must contain a single valid account ID and a single valid email address for the account that you want to add. Accounts must appear one per line, and the account ID and email address must be separated by a comma.

   ```
   Account ID,Email
   111111111111,user@example.com
   ```

5. When you are finished adding accounts, choose **Next**.

   The added accounts appear in a list on the **Accounts** page. Each added account in this list has an **Invite** link in the **Status** column.

#### Step 2 - Invite an account

2. In the navigation pane, choose **Accounts**.
3. For the account that you want to invite to enable GuardDuty, choose the **Invite** link that appears in the **Status** column of the added accounts list.
4. In the **Invitation to GuardDuty** dialog box, enter an invitation message (optional), and then choose **Send notification**.
Note
If the invited account does not have access to email, select Also send an email notification to the root user on the invitee's AWS account and generate an alert in the invitee's Personal Health Dashboard before sending the invitation.

The value in the Status column for the invited account changes to Pending.

Step 3 - Accept an invitation
2. Do one of the following:
   • If you don't have GuardDuty enabled, on the Enable GuardDuty page, choose Enable GuardDuty. Then use the Accept widget and the Accept invitation button to accept the membership invitation.
     Important
     You must enable GuardDuty before you can accept a membership invitation.
   • If you already have GuardDuty enabled, use the Accept widget and the Accept invitation button to accept the membership invitation.

After you accept the invitation, your account becomes a GuardDuty member account. The account whose user sent the invitation becomes the GuardDuty master account. The master account user can see that the value in the Status column for your member account changes to Monitored. The master account user can now view and manage GuardDuty findings for your member account.

Designating GuardDuty Master and Member Accounts Through Invitation (API)

You can designate master and member GuardDuty accounts by invitation through the API operations. Run the following GuardDuty API operations in order to designate master and member accounts in GuardDuty.

Complete the following procedure using the credentials of the AWS account that you want to designate as the GuardDuty master account.

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 master 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 master account by running the following CLI command. Make sure to use your own valid detector ID, account ID, and email.

aws guardduty create-members --detector-id 12abc34d567e8fa901bc2d34e56789f0 --account-details AccountId=123456789012,Email=guarddutymember@amazon.com

2. Run the InviteMembers API operation using the credentials of the AWS account that has GuardDuty enabled. This is the account that you want to be the master 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 using the `message` request parameter.

You can also use AWS Command Line Tools to designate member accounts by running the following CLI command. Make sure to use your own valid detector ID and valid account IDs for the accounts you want to invite.

```bash
aws guardduty invite-members --detector-id 12abc34d567e8fa901bc2d34e56789f0 --account-ids 123456789012
```

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.

   ```bash
   aws guardduty create-detector --enable
   ```

2. Run the `AcceptInvitation` 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 master account that sent the invitation, and the invitation ID of the invitation that you are accepting. You can find the account ID of the master 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, master account ID, and invitation ID.

   ```bash
   aws guardduty accept-invitation --detector-id 12abc34d567e8fa901bc2d34e56789f0 --master-id 012345678901 --invitation-id 84b097800250d17d1872b34c4daadcf5
   ```

**Enable GuardDuty in Multiple Accounts Simultaneously**

Use the following method to enable GuardDuty in multiple accounts at the same time.
Use Python Scripts to Enable GuardDuty in Multiple Accounts Simultaneously

You can automate the enabling or disabling of GuardDuty on multiple accounts using the scripts from the sample repository on GitHub at https://github.com/aws-samples/amazon-guardduty-multiaccount-scripts. Use the process in this section to enable GuardDuty for a list of member accounts using Amazon EC2. For information about using the disable script or setting up the script locally refer, to the GitHub instructions.

The enableguardduty.py script enables GuardDuty, sends invitations from the master account, and accepts invitations in all member accounts. The result is a master GuardDuty account that contains all security findings for all member accounts. Because GuardDuty is isolated by Region, findings for each member account roll up to the corresponding Region in the master account. For example, the us-east-1 Region in your GuardDuty master account contains the security findings for all us-east-1 findings from all associated member accounts.

These scripts have a dependency on a shared IAM role with the managed policy AmazonGuardDutyFullAccess. This policy provides entities access to GuardDuty and must be present on the master account and in each account for which you want to enable GuardDuty.

The following process enables GuardDuty in all available Regions by default. You can enable GuardDuty in specified Regions only by using the optional --enabled_regions argument and providing a comma-separated list of Regions. You can also optionally customize the invitation message that is sent to member accounts by opening the enableguardduty.py and editing the gd_invite_message string.

1. Create an IAM role in the GuardDuty master account and attach the AmazonGuardDutyFullAccess managed policy with the following permissions.

   ```json
   {
   "Version": "2012-10-17",
   "Statement": [
   {
   "Effect": "Allow",
   "Action": "guardduty:*",
   "Resource": "*"
   },
   {
   "Effect": "Allow",
   "Action": "iam:CreateServiceLinkedRole",
   "Resource": "*",
   "Condition": {
   "StringLike": {
   "iam:AWSServiceName": "guardduty.amazonaws.com"
   }
   }
   }
   ]
   }
   ```

2. Create an IAM role in each member account you want to be managed by your GuardDuty master account. This role must have the same name as the role created in step 1, it should allow the master account as a trusted entity, and it should have the same AmazonGuardDutyFullAccess managed policy described previously.

3. Launch a new Amazon Linux instance with an attached role that has the following trust relationship that allows the instance to assume a service role.

   ```json
   {
   }
4. Log in to the new instance and run the following commands to set it up.

```bash
sudo yum install git python
sudo yum install python-pip
pip install boto3
aws configure
git clone https://github.com/aws-samples/amazon-guardduty-multiaccount-scripts.git
cd amazon-guardduty-multiaccount-scripts
sudo chmod +x disableguardduty.py enableguardduty.py
```

5. Create a CSV file containing a list of account IDs and emails of the member accounts that you added a role to in step 2. Accounts must appear one per line, and the account ID and email address must be separated by a comma as in the following example.

```
111111111111,user@example.com
```

**Note**

The CSV file must be in the same location as your `enableguardduty.py` script. You can copy an existing CSV file from Amazon S3 to your current directory with the following method.

```
aws s3 cp s3://my-bucket/my_key_name example.csv
```

6. Run the Python script. Make sure to supply your GuardDuty master account ID, the name of the role created in the first steps, and the name of your CSV file as arguments.

```
python enableguardduty.py --master_account 111111111111 --assume_role roleName accountID.csv
```
Suspending or Disabling GuardDuty

You can use the GuardDuty console to suspend or disable GuardDuty. You are not charged for using GuardDuty when the service is suspended.

- All member accounts must be disassociated or deleted before you can disable or suspend 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.
- 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 disable GuardDuty.

To suspend or disable GuardDuty

2. In the navigation pane, under Settings, choose General.
3. Choose either Suspend GuardDuty or Disable GuardDuty. Then choose Save settings.
Monitoring GuardDuty Findings with Amazon CloudWatch Events

GuardDuty can send notifications based on Amazon CloudWatch Events when any changes in the findings take place. These changes include newly generated findings or subsequent occurrences of existing findings.

Every GuardDuty finding is assigned a finding ID. GuardDuty creates a CloudWatch event for every finding with a unique finding ID. All subsequent occurrences of an existing finding are always assigned a finding ID that is identical to the ID of the original finding.

In order to receive notifications about GuardDuty findings based on CloudWatch Events, you must create a CloudWatch Events rule and a target for GuardDuty. This rule enables CloudWatch to send events for all findings that GuardDuty generates to the target that is specified in the rule. For more information, see Creating a CloudWatch Events Rule and Target for GuardDuty (p. 81).

Topics
- CloudWatch Events Notification Frequency for GuardDuty (p. 80)
- Monitoring Archived GuardDuty Findings with CloudWatch Events (p. 81)
- CloudWatch Event Format for GuardDuty (p. 81)
- Creating a CloudWatch Events Rule and Target for GuardDuty (p. 81)

CloudWatch Events Notification Frequency for GuardDuty

Notifications for newly generated findings with a unique finding ID – GuardDuty sends a notification based on its CloudWatch event within 5 minutes of the finding. This event (and this notification) also includes all subsequent occurrences of this finding that take place in the first 5 minutes since this finding with a unique ID is generated.

Important
You CANNOT customize the default frequency (5 minutes) of notifications sent about the newly generated findings.

Notifications for subsequent finding occurrences – By default, for every finding with a unique finding ID, GuardDuty aggregates all subsequent occurrences of a particular finding that take place in 6-hour intervals into a single event. GuardDuty then sends a notification about these subsequent occurrences based on this event. In other words, for the subsequent occurrences of the existing findings, GuardDuty sends notifications based on CloudWatch events every 6 hours.

Important
You can customize the default frequency of notifications sent about the subsequent finding occurrences. Possible values are 15 minutes, 1 hour, or the default 6 hours. You can update this value using the CreateDetector or the UpdateDetector API operations. You can also update this value through the GuardDuty console - choose Settings and then under CloudWatch Events, choose one of the available values from the Updated findings pull-down menu. Only users from a master account can customize the default frequency of notifications sent about the subsequent finding occurrences to CloudWatch Events. Users from member accounts...
CANNOT customize this frequency value. The frequency value set by the master account in its own account is imposed on GuardDuty functionality in all its member accounts. In other words, if a user from a master account sets this frequency value to 1 hour, all member accounts will also have the 1 hour frequency of notifications about the subsequent finding occurrences sent to CloudWatch Events. For more information, see Managing Multiple Accounts in Amazon GuardDuty (p. 68).

Monitoring Archived GuardDuty Findings with CloudWatch Events

For the manually archived findings, the initial and all subsequent occurrences of these finding (generated after the archiving is complete) are sent to CloudWatch Events per frequency described above.

For the auto-archived findings, the initial and all subsequent occurrences of these findings (generated after the archiving is complete) are not sent to CloudWatch Events.

CloudWatch Event Format for GuardDuty

The CloudWatch event for GuardDuty has the following format.

```
{
  "version": "0",
  "id": "cd3d702e-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": {COMPLETE_GUARDDUTY_FINDING_JSON}
}
```

For the complete list of all the parameters included in COMPLETE_GUARDDUTY_FINDING_JSON, see GetFindings. The id parameter that appears in COMPLETE_GUARDDUTY_FINDING_JSON is the finding ID previously described.

Creating a CloudWatch Events Rule and Target for GuardDuty

The following procedure shows how to use AWS CLI commands to create a CloudWatch Events rule and target for GuardDuty. Specifically, the procedure shows you how to create a rule that enables CloudWatch to send events for all findings that GuardDuty generates and add an AWS Lambda function as a target for the rule.

**Note**
In addition to Lambda functions, GuardDuty and CloudWatch 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.
You can also create a CloudWatch Events rule and target for GuardDuty through the CloudWatch Events console. For more information and detailed steps, see Creating a CloudWatch Events Rule That Triggers on an Event. In the Event Source section, select GuardDuty for Service name and GuardDuty Finding for Event Type.

To create a rule and target

1. To create a rule that enables CloudWatch to send events for all findings that GuardDuty generates, run the following CloudWatch CLI command.

   ```bash
   aws events put-rule --name Test --event-pattern "{"source": ["aws.guardduty"]}"
   ```

   **Important**
   You can further customize your rule so that it instructs CloudWatch 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 CloudWatch to only send events for the GuardDuty findings with the severity of either 5 or 8:

   ```bash
   aws events put-rule --name Test --event-pattern "{"source": ["aws.guardduty"], "detail-type": ["GuardDuty Finding"], "detail": {"severity": [5, 8]}}"
   ```

   For this purpose, you can use any of the GuardDuty attributes that are supported for sorting findings. For more information, see GetFindings.

2. To attach a Lambda function as a target for the rule that you created in step 1, run the following CloudWatch CLI command.

   ```bash
   aws events put-targets --rule Test --targets Id=1,Arn=arn:aws:lambda:us-east-1:111122223333:function:<your_function>
   ```

   **Note**
   Make sure to replace `<your_function>` 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.

   ```bash
   aws lambda add-permission --function-name <your_function> --statement-id 1 --action 'lambda:InvokeFunction' --principal events.amazonaws.com
   ```

   **Note**
   Make sure to replace `<your_function>` in the command above with your actual Lambda function for the GuardDuty events.

   **Note**
   In the procedure above, we’re using a Lambda function as the target for the rule that triggers CloudWatch Events. You can also configure other AWS resources as targets to trigger CloudWatch Events. For more information, see PutTargets.
Subscribing to GuardDuty Announcements SNS Topic

This section provides information on subscribing 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.

**Note**
Your user account must have sns::subscribe IAM permissions to subscribe to an SNS topic.

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 Notification Service Developer Guide.

The Amazon SNS topic ARNs for each Region are shown below.

<table>
<thead>
<tr>
<th>AWS Region</th>
<th>Amazon SNS Topic ARN</th>
</tr>
</thead>
<tbody>
<tr>
<td>us-east-1</td>
<td>arn:aws:sns:us-east-1:242987662583:GuardDutyAnnouncements</td>
</tr>
<tr>
<td>us-east-2</td>
<td>arn:aws:sns:us-east-2:118283430703:GuardDutyAnnouncements</td>
</tr>
<tr>
<td>us-west-1</td>
<td>arn:aws:sns:us-west-1:144182107116:GuardDutyAnnouncements</td>
</tr>
<tr>
<td>ca-central-1</td>
<td>arn:aws:sns:ca-central-1:107430051933:GuardDutyAnnouncements</td>
</tr>
<tr>
<td>eu-north-1</td>
<td>arn:aws:sns:eu-north-1:973841112453:GuardDutyAnnouncements</td>
</tr>
<tr>
<td>eu-west-1</td>
<td>arn:aws:sns:eu-west-1:965013871422:GuardDutyAnnouncements</td>
</tr>
<tr>
<td>eu-west-2</td>
<td>arn:aws:sns:eu-west-2:506403581195:GuardDutyAnnouncements</td>
</tr>
<tr>
<td>eu-west-3</td>
<td>arn:aws:sns:eu-west-3:436163563069:GuardDutyAnnouncements</td>
</tr>
<tr>
<td>eu-central-1</td>
<td>arn:aws:sns:eu-central-1:378365507264:GuardDutyAnnouncements</td>
</tr>
</tbody>
</table>
AWS Region | Amazon SNS Topic ARN
--- | ---
ap-east-1 | arn:aws:sns:ap-east-1:646602203151:GuardDutyAnnouncements
ap-northeast-1 | arn:aws:sns:ap-northeast-1:741172661024:GuardDutyAnnouncements
ap-southeast-1 | arn:aws:sns:ap-southeast-1:476419727788:GuardDutyAnnouncements
ap-south-1 | arn:aws:sns:ap-south-1:926826061926:GuardDutyAnnouncements
sa-east-1 | arn:aws:sns:sa-east-1:955633302743:GuardDutyAnnouncements

To subscribe to the GuardDuty update notification email in the AWS Management Console

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.
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:

```bash
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.
An example GuardDuty update notification message about new findings is shown below:

```
{
   "Type" : "Notification",
   "MessageId" : "9101dc6b-726f-4df0-8646-ec2f94e674bc",
   "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" : "XWox8GDLRiCgDOXlo/fG9Lu/88P6Q6F6L6o6QyOuMFiuzokuhol15e3BqjdCHeqKt7qdpHMPQ0LpN7y9iBrWVUqAGJrukiA8athvAS+4AQD/VQjhsEnlj+GaiW+ozAu006XGopOsFgOCoFtPIMOjCMrMonjz7HPv/8KRUZRT3pyQYm5d4WB7XMPyhU4wLZ1V8YFs55FMtgQV/YLh1SYv6uOP1MTQauxKpoeCotFFP/VhQQLFx1q9LR0dCCiRHiTw0XWl87PS1+BVk1in6AL7PhkvdQ7FyAhxFxM5+6p8Q0yKCqeeBG7HN3KlAbbyVka7SNRO/6asyr1jg==",
}
```

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",
}
```
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."
  }
}
```

An example GuardDuty update notification message about updated findings is shown below:

```
{  
  "Type": "Notification",
  "MessageId": "9101dc6b-726f-4df0-8646-ec2f94e674bc",
  "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."
      }
    ]  
  }
}
```

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."
  }
}
"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."
}]
}
Quotas for Amazon GuardDuty

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 cannot 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.

<table>
<thead>
<tr>
<th>Resource</th>
<th>Default</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>Detectors</td>
<td>1</td>
<td>The maximum number of detector resources that you can create per AWS account per Region. You cannot request a quota increase.</td>
</tr>
<tr>
<td>Filters</td>
<td>100</td>
<td>The maximum number of saved filters per AWS account per Region.</td>
</tr>
<tr>
<td>Finding retention period</td>
<td>90 days</td>
<td>The maximum number of days a finding is retained. You cannot request a quota increase.</td>
</tr>
<tr>
<td>IP addresses and CIDR ranges per Trusted IP List</td>
<td>2,000</td>
<td>The maximum number of IP addresses and CIDR ranges that you can include in a single Trusted IP List. You cannot request a quota increase.</td>
</tr>
<tr>
<td>IP addresses and CIDR ranges per Threat List</td>
<td>250,000</td>
<td>The maximum number of IP address and CIDR ranges that you can include in a Threat List. You cannot request a quota increase.</td>
</tr>
<tr>
<td>Maximum file size</td>
<td>35 MB</td>
<td>The maximum file size for the file used to upload a list of IP addresses and CIDR ranges.</td>
</tr>
<tr>
<td>Resource</td>
<td>Default</td>
<td>Comments</td>
</tr>
<tr>
<td>---------------------</td>
<td>---------</td>
<td>---------------------------------------------------------------------------</td>
</tr>
<tr>
<td></td>
<td></td>
<td>addresses or CIDR ranges to include in a Trusted IP List or a Threat List.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>You cannot request a quota increase.</td>
</tr>
<tr>
<td>Member accounts</td>
<td>5000</td>
<td>The maximum number of member accounts associated with a master account.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>You can have one master account per detector.</td>
</tr>
<tr>
<td>Threat intel sets</td>
<td>6</td>
<td>The maximum number of Threat intel sets that you can add per AWS account.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>per Region.</td>
</tr>
<tr>
<td>Trusted IP sets</td>
<td>1</td>
<td>The maximum number of trusted IP sets that can be uploaded and activated</td>
</tr>
<tr>
<td></td>
<td></td>
<td>per AWS account per Region.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>You cannot request a quota increase.</td>
</tr>
</tbody>
</table>
Regions and Endpoints

To view the Regions where Amazon GuardDuty is available, see Amazon GuardDuty Endpoints in the Amazon Web Services General Reference.

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 allows GuardDuty to monitor AWS CloudTrail events for global AWS services such as AWS Identity and Access Management (IAM). If GuardDuty is not enabled in all supported Regions, its ability to detect activity that involves global services is reduced.

If you are using GuardDuty in AWS GovCloud (US), there are differences. For more information, see Amazon GuardDuty in the AWS GovCloud (US) User Guide.
# Document History for Amazon GuardDuty

<table>
<thead>
<tr>
<th>update-history-change</th>
<th>update-history-description</th>
<th>update-history-date</th>
</tr>
</thead>
<tbody>
<tr>
<td>Added content for AWS Organizations integration.</td>
<td>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 master you can automatically enable GuardDuty for any organization member to be managed by the master account. You can also automatically enable GuardDuty in new AWS Organizations member accounts. <a href="#">Learn more.</a></td>
<td>April 20, 2020</td>
</tr>
<tr>
<td>Added content for the Export Findings feature.</td>
<td>Added content that describes the Export Findings feature of GuardDuty.</td>
<td>November 14, 2019</td>
</tr>
<tr>
<td>Added the UnauthorizedAccess:EC2/MetadataDNSRebind finding type.</td>
<td>Added a new Unauthorized finding, UnauthorizedAccess:EC2/MetadataDNSRebind. <a href="#">Learn more.</a></td>
<td>October 10, 2019</td>
</tr>
<tr>
<td>Added the Stealth:IAMUser/S3ServerAccessLoggingDisabled finding type.</td>
<td>Added a new Stealth finding, Stealth:IAMUser/S3ServerAccessLoggingDisabled. <a href="#">Learn more.</a></td>
<td>October 10, 2019</td>
</tr>
<tr>
<td>Added the Policy:IAMUser/S3BlockPublicAccessDisabled finding type.</td>
<td>Added a new Policy finding, Policy:IAMUser/S3BlockPublicAccessDisabled. <a href="#">Learn more.</a></td>
<td>October 10, 2019</td>
</tr>
<tr>
<td>Retired the Backdoor:EC2/XORDDOS finding type.</td>
<td>The Backdoor:EC2/XORDDOS finding type is now retired from GuardDuty. <a href="#">Learn more</a></td>
<td>June 12, 2019</td>
</tr>
<tr>
<td>Added the PrivilegeEscalation finding type.</td>
<td>The PrivilegeEscalation finding type detects when users attempt to assign escalated, more permissive privileges to their accounts. <a href="#">Learn more</a></td>
<td>May 14, 2019</td>
</tr>
<tr>
<td>GuardDuty is now available in the Europe (Stockholm) Region.</td>
<td>Added Europe (Stockholm) to the list of AWS Regions in which GuardDuty is available. <a href="#">Learn more</a></td>
<td>May 9, 2019</td>
</tr>
<tr>
<td>Added a new finding type, Recon:EC2/PortProbe:EMRUnprotectedPort.</td>
<td>GuardDuty is available. Learn more</td>
<td>May 8, 2019</td>
</tr>
<tr>
<td>Added 5 new finding types that detect if your EC2 instances are potentially being used for Denial of Service (DoS) attacks.</td>
<td>This finding informs you that an EMR-related sensitive port on an EC2 Instance is not blocked and is being actively probed. Learn more</td>
<td>March 8, 2019</td>
</tr>
<tr>
<td>Added a new finding type: Policy:IAMUser/RootCredentialUsage</td>
<td>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</td>
<td>January 24, 2019</td>
</tr>
<tr>
<td>UnauthorizedAccess:IAMUser/UnusualASNCaller finding type has been retired</td>
<td>Policy:IAMUser/RootCredentialUsage finding type informs you that the root credentials of your AWS account are being used to make programmatic requests to AWS services. Learn more</td>
<td>December 21, 2018</td>
</tr>
<tr>
<td>Added two new finding types: PenTest:IAMUser/ParrotLinux and PenTest:IAMUser/PentooLinux</td>
<td>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</td>
<td>December 21, 2018</td>
</tr>
<tr>
<td></td>
<td>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</td>
<td></td>
</tr>
<tr>
<td>Added support for the Amazon GuardDuty Announcements SNS Topic</td>
<td>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. <a href="#">Learn more</a></td>
<td>November 21, 2018</td>
</tr>
<tr>
<td>---</td>
<td>---</td>
<td>---</td>
</tr>
<tr>
<td>Added two new finding types: UnauthorizedAccess:EC2/TorClient and UnauthorizedAccess:EC2/TorRelay</td>
<td>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. <a href="#">Learn more</a></td>
<td>November 16, 2018</td>
</tr>
<tr>
<td>Added a new finding type: CryptoCurrency:EC2/BitcoinTool.B</td>
<td>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. <a href="#">Learn more</a></td>
<td>November 9, 2018</td>
</tr>
<tr>
<td>Added support for updating the frequency of notifications sent to CloudWatch Events</td>
<td>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. <a href="#">Learn more</a></td>
<td>October 9, 2018</td>
</tr>
<tr>
<td>Added Region support</td>
<td>Added Region support for AWS GovCloud (US-West) <a href="#">Learn more</a></td>
<td>July 25, 2018</td>
</tr>
<tr>
<td>Added support for AWS CloudFormation StackSets in GuardDuty</td>
<td>You can use the Enable Amazon GuardDuty template to enable GuardDuty simultaneously in multiple accounts. <a href="#">Learn more</a></td>
<td>June 25, 2018</td>
</tr>
</tbody>
</table>
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 continuous security monitoring and threat detection in this Region. Learn more | March 29, 2018

Creating GuardDuty master 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 (GuardDuty master account). Learn more | January 25, 2018

Changes in upload and further management of trusted IP lists and threat lists for GuardDuty master and member accounts. | With this release, Users from master GuardDuty accounts can upload and manage trusted IP lists and threat lists. Users from member GuardDuty accounts CANNOT upload and manage lists. Trusted IP lists and threat lists that are uploaded by the master account are imposed on GuardDuty functionality in its member accounts. Learn more | January 25, 2018

The following table describes important changes in each release of the GuardDuty User Guide.
## Earlier updates

<table>
<thead>
<tr>
<th>Change</th>
<th>Description</th>
<th>Date</th>
</tr>
</thead>
<tbody>
<tr>
<td>Initial publication</td>
<td>Initial publication of the Amazon GuardDuty User Guide.</td>
<td>November 28, 2017</td>
</tr>
</tbody>
</table>