



User Guide

Amazon Inspector



Amazon Inspector: User Guide

Copyright © 2024 Amazon Web Services, Inc. and/or its affiliates. All rights reserved.

Amazon's trademarks and trade dress may not be used in connection with any product or service that is not Amazon's, in any manner that is likely to cause confusion among customers, or in any manner that disparages or discredits Amazon. All other trademarks not owned by Amazon are the property of their respective owners, who may or may not be affiliated with, connected to, or sponsored by Amazon.

Table of Contents

What is Amazon Inspector?	1
Features	1
Accessing Amazon Inspector	3
Getting started	5
Before activating Amazon Inspector	5
Getting started tutorial: Activating Amazon Inspector	6
Automated scans	8
Overview of Amazon Inspector scan types	8
Activating a scan type	10
Activating scans	11
Scanning Amazon EC2 instances	11
Agent-based scanning	12
Agentless scanning	16
Managing scan mode	18
Excluding instances from Amazon Inspector scans	19
Supported operating systems	19
Deep inspection for Linux instances	19
Scanning Windows EC2 instance	25
Scanning Amazon ECR container images	28
Scan behaviors for Amazon ECR scanning	29
Supported operating systems and media types	30
Configuring the Amazon ECR re-scan duration	30
Scanning Lambda functions	32
Scan behaviors for Lambda function scanning	33
Supported runtimes and functions	33
Amazon Inspector Lambda standard scanning	34
Amazon Inspector Lambda code scanning	35
Deactivating a scan type	37
Deactivating scans	38
CIS scans	40
Amazon EC2 instance requirements for Amazon Inspector CIS scans	41
Amazon Virtual Private Cloud endpoint requirements for running CIS scans on private	
Amazon EC2 instances	41
Running CIS scans	42

Considerations for managing Amazon Inspector CIS scans with AWS Organizations	43
Amazon Inspector owned Amazon S3 buckets used for Amazon Inspector CIS scans	44
Creating a CIS scan configuration	46
Viewing CIS scan results	47
Editing a CIS scan configuration	48
Downloading a CIS scan results	49
Understanding findings	50
Finding types	51
Package vulnerability	51
Code vulnerability	51
Network reachability	52
Viewing findings	53
Viewing finding details	54
Viewing the Amazon Inspector score	58
Amazon Inspector score	58
Vulnerability Intelligence	60
Understanding severity levels for findings	61
Software package vulnerability severity	61
Code vulnerability severity	62
Network reachability severity	61
Managing findings	65
Filtering findings	65
Creating filters in the Amazon Inspector console	65
Suppressing findings	66
Creating a suppression rule	67
Viewing suppressed findings	67
Changing suppression rules	68
Deleting suppression rules	68
Exporting findings reports	68
Step 1: Verify your permissions	69
Step 2: Configure an S3 bucket	71
Step 3: Configure an AWS KMS key	74
Step 4: Configure and export a findings report	78
Troubleshoot errors	80
Automating responses to findings with EventBridge	81
Event schema	81

Creating an EventBridge rule to notify you of Amazon Inspector findings	84
EventBridge for Amazon Inspector multi-account environments	88
Dashboard	89
Viewing the dashboard	89
Understanding dashboard components	90
Searching the vulnerability database	93
Searching the vulnerability database	93
Understanding CVE details	94
CVE details	94
Vulnerability intelligence	94
References	94
Exporting SBOMs	95
Amazon Inspector formats	95
Filters for SBOMs	100
Configure and export SBOMs	101
EventBridge schema	103
Amazon EventBridge base schema for Amazon Inspector	103
Amazon Inspector finding event schema example	104
Amazon Inspector initial scan complete event schema example	116
Amazon Inspector coverage event schema example	119
CI/CD integration	121
Plugin integration	121
Supported CI/CD solutions	122
Custom integration	122
Set up an account for CI/CD integration	123
Sign up for an AWS account	124
Create a user with administrative access	124
Configure an IAM role for CI/CD integration	125
Amazon Inspector SBOM Generator	127
Supported packages types	127
Supported container image configuration checks	128
Installing S bomgen	128
Using S bomgen	129
Authenticating to private registries with S bomgen	134
Example outputs from S bomgen	135
Previous versions	137

Amazon Inspector Dockerfile checks	139
Using Sbomgen Dockerfile checks	140
Supported Dockerfile checks	142
Creating a custom CI/CD integration	146
Step 1. Configuring AWS account	147
Step 2. Installing Sbomgen binary	147
Step 3. Using Sbomgen	147
Step 4. Calling the Amazon Inspector Scan API	147
(Optional) Step 5. Generate and scan SBOM in a single command	148
API output formats	148
Jenkins plugin	156
Step 1. Set up an AWS account	157
Step 2. Install the Amazon Inspector Jenkins Plugin	157
(Optional) Step 3. Add docker credentials to Jenkins	157
(Optional) Step 4. Add AWS credentials	158
Step 5. Add CSS support in a Jenkins script	158
Step 6. Add Amazon Inspector Scan to your build	158
Step 7. View your Amazon Inspector vulnerability report	162
Troubleshooting	162
TeamCity plugin	164
GitHub actions	166
GitLab components	166
Using CodeCatalyst actions	167
Using CycloneDX namespaces	167
amazon:inspector:sbom_scanner namespace taxonomy	167
amazon:inspector:sbom_generator namespace taxonomy	169
Assessing coverage	171
Assessing account-level coverage	172
Assessing coverage of Amazon EC2 instances	172
Amazon EC2 instances status values	173
Assessing coverage of Amazon ECR repositories	175
Amazon ECR repository scan status values	176
Assessing coverage of Amazon ECR container images	177
Amazon ECR container image scan status values	177
Assessing coverage of AWS Lambda functions	179
Lambda functions scan status values	179

Managing multiple accounts	181
Understanding the delegated administrator account and member account	181
Delegated administrator actions	181
Member account actions	183
Designating an administrator account	183
Important considerations for delegated administrators	184
Permissions required to designate a delegated administrator	184
Designating a delegated administrator	185
Activating Amazon Inspector scans for member accounts	187
Disassociating member accounts	190
Removing the delegated administrator	191
Usage	193
Using the usage console	193
Understanding how Amazon Inspector calculates usage costs	195
About the Amazon Inspector free trial	195
Security	196
Data protection	197
Encryption at rest	198
Encryption in transit	202
Identity and Access Management	202
Audience	203
Authenticating with identities	203
Managing access using policies	207
How Amazon Inspector works with IAM	209
Identity-based policy examples	215
AWS managed policies	220
Using service-linked roles	231
Troubleshooting	245
Monitoring Amazon Inspector	247
CloudTrail logs	248
Compliance validation	251
Resilience	252
Infrastructure security	252
Incident response	253
Integrations	254
Integrating Amazon Inspector with Amazon ECR	254

Amazon Inspector integration with Security Hub	254
Amazon ECR integration	254
Activating the integration	255
Using the integration with a multi-account environment	255
Security Hub integration	255
Viewing Amazon Inspector findings in AWS Security Hub	256
Activating and configuring the Amazon Inspector integration with Security Hub	259
Disabling the flow of findings from an integration	260
Viewing security controls for Amazon Inspector in Security Hub	260
Supported operating systems and programming languages	261
Supported operating systems	262
Supported operating systems: Amazon EC2 scanning	262
Supported operating systems: Amazon ECR scanning with Amazon Inspector	265
Supported operating systems: CIS scanning	267
Discontinued operating systems	268
Supported programming languages	273
Supported programming languages: Amazon EC2 agentless scanning	273
Supported programming languages: Amazon EC2 deep inspection	274
Supported programming languages: Amazon ECR scanning	274
Supported runtimes	275
Supported runtimes: Amazon Inspector Lambda standard scanning	275
Supported runtimes: Amazon Inspector Lambda code scanning	276
Deactivating Amazon Inspector	278
Deactivate Amazon Inspector	279
Quotas	280
Regions and endpoints	282
Endpoints for Amazon Inspector Scan API	282
Region-specific feature availability	286
Document history	289
AWS Glossary	303

What is Amazon Inspector?

Amazon Inspector is a vulnerability management service that automatically discovers workloads and continually scans them for software vulnerabilities and unintended network exposure. Amazon Inspector discovers and scans [Amazon EC2 instances](#), [container images in Amazon ECR](#), and [Lambda functions](#). When Amazon Inspector detects a software vulnerability or unintended network exposure, it creates [a finding](#), which is a detailed report about the issue. You can [manage findings](#) in the Amazon Inspector console or API.

Topics

- [Features of Amazon Inspector](#)
- [Accessing Amazon Inspector](#)

Features of Amazon Inspector

Centrally manage multiple Amazon Inspector accounts

If your AWS environment has multiple accounts, you can centrally manage your environment through a single account by using AWS Organizations. Using this approach, you can designate an account as the delegated administrator account for Amazon Inspector.

Amazon Inspector can be activated for your entire organization with a single click. Additionally, you can automate activating the service for future members whenever they join your organization. The Amazon Inspector delegated administrator account can manage findings data and certain settings for members of the organization. This includes viewing aggregated findings details for all member accounts, activating or deactivating scans for member accounts, and reviewing scanned resources within the AWS organization.

Continuously scan your environment for vulnerabilities and network exposure

With Amazon Inspector, you don't need to manually schedule or configure assessment scans. Amazon Inspector automatically discovers and begins [scanning your eligible resources](#). Amazon Inspector continues to assess your environment throughout the lifecycle of your resources by automatically rescanning resources in response to changes that could introduce a new vulnerability, such as: installing a new package in an EC2 instance, installing a patch, and when a new common vulnerabilities and exposures (CVE) that impacts the resource is published. Unlike

traditional security scanning software, Amazon Inspector has minimal impact on the performance of your fleet.

When vulnerabilities or open network paths are identified, Amazon Inspector produces a [finding](#) that you can investigate. The finding includes comprehensive details about the vulnerability, the affected resource, and remediation recommendations. If you appropriately remediate a finding, Amazon Inspector automatically detects the remediation and closes the finding.

Assess vulnerabilities accurately with the Amazon Inspector Risk score

As Amazon Inspector collects information about your environment through scans, it provides severity scores specifically tailored to your environment. Amazon Inspector examines the security metrics that compose the [National Vulnerability Database \(NVD\)](#) base score for a vulnerability and adjusts them according to your compute environment. For example, the service may lower the Amazon Inspector score of a finding for an Amazon EC2 instance if the vulnerability is exploitable over the network but no open network path to the internet is available from the instance. This score is in CVSS format and is a modification of the base [Common Vulnerability Scoring System \(CVSS\)](#) score provided by NVD.

Identify high-impact findings with the Amazon Inspector dashboard

The [Amazon Inspector dashboard](#) offers a high-level view of findings from across your environment. From the dashboard, you can access the granular details of a finding. The dashboard contains streamlined information about scan coverage in your environment, your most critical findings, and which resources have the most findings. The risk-based remediation panel in the Amazon Inspector dashboard presents the findings that affect the largest number of instances and images. This panel makes it easier to identify the findings with the greatest impact on your environment, review finding details, and review suggested solutions.

Manage your findings using customizable views

In addition to the dashboard, the Amazon Inspector console offers a **Findings** view. This page lists all findings for your environment and provides the details of individual findings. You can view findings grouped by category or vulnerability type. In each view, you can further customize your results using filters. You can also use filters to create suppression rules that hide unwanted findings from your views.

You can use filters and suppression rules to generate finding reports that show all findings or a customized selection of findings. Reports can be generated in CSV or JSON formats.

Monitor and process findings with other services and systems

To support integration with other services and systems, Amazon Inspector [publishes findings to Amazon EventBridge](#) as finding events. EventBridge is a serverless event bus service that can route findings data to targets such as AWS Lambda functions and Amazon Simple Notification Service (Amazon SNS) topics. With EventBridge, you can monitor and process findings in near-real time as part of your existing security and compliance workflows.

If you have activated [AWS Security Hub](#), then Amazon Inspector will also [publish findings to Security Hub](#). Security Hub is a service that provides a comprehensive view of your security posture across your AWS environment and helps you check your environment against security industry standards and best practices. With Security Hub, you can more easily monitor and process your findings as part of a broader analysis of your organization's security posture in AWS.

Accessing Amazon Inspector

Amazon Inspector is available in most AWS Regions. For a list of Regions where Amazon Inspector is currently available, see [Amazon Inspector endpoints and quotas](#) in the *Amazon Web Services General Reference*. To learn more about AWS Regions, see [Managing AWS Regions](#) in the *Amazon Web Services General Reference*. In each Region, you can work with Amazon Inspector in the following ways.

AWS Management Console

The AWS Management Console is a browser-based interface that you can use to create and manage AWS resources. As part of that console, the Amazon Inspector console provides access to your Amazon Inspector account and resources. You can perform Amazon Inspector tasks from the Amazon Inspector console.

AWS command line tools

With AWS command line tools, you can issue commands at your system's command line to perform Amazon Inspector tasks. Using the command line can be faster and more convenient than using the console. The command line tools are also useful if you want to build scripts that perform tasks.

AWS provides two sets of command line tools: the AWS Command Line Interface (AWS CLI) and the AWS Tools for PowerShell. For information about installing and using the AWS CLI, see the [AWS Command Line Interface User Guide](#). For information about installing and using the Tools for PowerShell, see the [AWS Tools for PowerShell User Guide](#).

AWS SDKs

AWS provides SDKs that consist of libraries and sample code for various programming languages and platforms, including Java, Go, Python, C++, and .NET. The SDKs provide convenient, programmatic access to Amazon Inspector and other AWS services. They also handle tasks such as cryptographically signing requests, managing errors, and retrying requests automatically. For information about installing and using the AWS SDKs, see [Tools to Build on AWS](#).

Amazon Inspector REST API

The Amazon Inspector REST API gives you comprehensive, programmatic access to your Amazon Inspector account and resources. With this API, you can send HTTPS requests directly to Amazon Inspector. However, unlike the AWS command line tools and SDKs, use of this API requires your application to handle low-level details such as generating a hash to sign a request.

Getting started with Amazon Inspector

This section provides information to consider before activating Amazon Inspector and a getting started tutorial describing how to activate Amazon Inspector and view your [findings](#) in the Amazon Inspector console and with the Amazon Inspector API.

Topics

- [Before activating Amazon Inspector](#)
- [Getting started tutorial: Activating Amazon Inspector](#)

Before activating Amazon Inspector

Before activating Amazon Inspector, consider the following:

Amazon Inspector is a Regional service

Your data is stored in the AWS Region where you activate Amazon Inspector. Repeat the steps in the first part of the [getting started tutorial](#) for all AWS Regions where you plan to use Amazon Inspector.

Amazon Inspector creates the service-linked roles `AWSServiceRoleForAmazonInspector2` and `AWSServiceRoleForAmazonInspector2Agentless`

A [service-linked role](#) is a role in AWS Identity and Access Management (IAM) that's linked to an AWS service. [AWSServiceRoleForAmazonInspector2](#) and [AWSServiceRoleForAmazonInspector2Agentless](#) allow Amazon Inspector to access AWS services required to perform security assessments.

IAM identities with administrator permissions can enable Amazon Inspector

Protect your credentials by creating users with [IAM](#) or [AWS IAM Identity Center](#). This helps you make sure users only have the permissions required to manage Amazon Inspector. For more information, see [AWS managed policy: AmazonInspectorFullAccess](#).

Hybrid scanning is automatically enabled

Hybrid scanning includes [agent-based scanning](#) and [agentless scanning](#). By default, Amazon Inspector uses these scan methods on all eligible Amazon EC2 instances. For more information, see [Scanning Amazon EC2 instances with Amazon Inspector](#).

Amazon ECR scanning and Lambda function scanning doesn't require the SSM agent

Agent-based scanning uses [the SSM agent](#) to collect software inventory. Agentless scanning uses Amazon EBS snapshots to collect software inventory.

Note

By default, the SSM agent is already installed in Amazon EC2 instances based on Amazon Machine Images. However, you might need to activate the SSM agent manually in some cases. For more information, see [Working with the SSM agent](#) in the *AWS Systems Manager User Guide*.

Monthly costs are based on workloads scanned

For more information, see [Amazon Inspector pricing](#).

Getting started tutorial: Activating Amazon Inspector

This getting started tutorial describes how to activate Amazon Inspector for a standalone account environment (member account) and multi-account environment (delegated administrator account). When you activate Amazon Inspector, it automatically begins discovering workloads and scanning them for software vulnerabilities and unintended network exposure. For information about the delegated administrator account and member accounts, see [Understanding the delegated administrator account and member accounts in Amazon Inspector](#).

Standalone account environment

This procedure describes how to activate Amazon Inspector in the console. To programmatically activate Amazon Inspector, use the Amazon Inspector shell script. For more information, see the [inspector2-enablement-with-cli](#) on GitHub.

1. Sign in using your credentials, and then open the Amazon Inspector console at <https://console.aws.amazon.com/inspector/v2/home>.
2. Choose **Get Started**.
3. Choose **Activate Amazon Inspector**.

When you activate Amazon Inspector for a standalone account, all scan types are activated by default. For more information, see [Automated scan types for Amazon Inspector](#).

Multi-account environment

This procedure describes how to activate Amazon Inspector in the console. To programmatically activate Amazon Inspector, use the Amazon Inspector shell script. For more information, see the [inspector2-enablement-with-cli](#) on GitHub.

Note

To complete this procedure, you must have access to the AWS Organizations management account and be in the same organization as the accounts you want to manage. Permissions might be required to delegate an administrator. For more information, see [Permissions required to designate a delegated administrator](#).

Delegating an administrator for Amazon Inspector

1. Sign in to the AWS Organizations management account, and then open the Amazon Inspector console at <https://console.aws.amazon.com/inspector/v2/home>.
2. For **Delegated administrator account ID**, enter the 12-digit ID of the AWS account you want to designate as the delegated administrator, and then choose **Delegate**.
3. In the confirmation window, choose **Delegate** again.

When you activate Amazon Inspector for a delegated administrator, all scan types are activated by default. For more information, see [Automated scan types for Amazon Inspector](#).

Automated scan types in Amazon Inspector

Amazon Inspector uses a purpose-built scanning engine that monitors your resources for software vulnerabilities and unintended network exposure. When Amazon Inspector detects a software vulnerability or unintended network exposure, it creates a [finding](#). When you activate Amazon Inspector for the first time, your account is automatically enrolled in [all scan types](#), which include Amazon Amazon EC2 scanning, Amazon ECR Scanning, and Lambda standard scanning.

Note

Lambda code scanning is an optional layer of Lambda function scanning that you can activate at any time.

Topics

- [Overview of Amazon Inspector scan types](#)
- [Activating a scan type](#)
- [Scanning Amazon EC2 instances with Amazon Inspector](#)
- [Scanning Amazon Elastic Container Registry container images with Amazon Inspector](#)
- [Scanning AWS Lambda functions with Amazon Inspector](#)
- [Deactivating a scan type in Amazon Inspector](#)

Overview of Amazon Inspector scan types

Amazon Inspector offers different scan types that focus on specific resource types in your AWS environment.

Amazon EC2 scanning

When you activate Amazon EC2 scanning, Amazon Inspector scans your EC2 instances for the following:

- Common vulnerabilities and exposures
- Operating system and programming language package vulnerabilities
- Network reachability

- Network exposure issues

Amazon Inspector performs scans through the use of the SSM agent installed on your instance or through Amazon EBS snapshots of instances. For more information about scans for Amazon EC2, see [Scanning Amazon EC2 instances with Amazon Inspector](#).

Note

By default, when you activate Amazon EC2 scanning, you automatically enable hybrid scanning mode. For more information, see [Agentless scanning](#).

Amazon ECR scanning

When you activate Amazon ECR scanning, Amazon Inspector converts all **Basic scanning** container repositories in your private registry to **Enhanced scanning** with continual scanning. You can also optionally configure this setting to scan on-push only or to scan select repositories through inclusion rules. All images pushed within the last 30 days, or pulled within the last 90 days are initially scanned. Amazon Inspector continues to monitor images for a 90 day duration by default, this setting can be changed at any time. For more information about scans for Amazon ECR, see [Scanning Amazon Elastic Container Registry container images with Amazon Inspector](#).

Lambda standard scanning

When you activate Lambda standard scanning, Amazon Inspector discovers the Lambda functions in your account and immediately starts scanning them for vulnerabilities. Amazon Inspector scans new Lambda functions and layers when they're deployed, and rescans them when they're updated or when new Common Vulnerabilities and Exposures (CVEs) are published. For more information about Lambda function scanning, see [Scanning AWS Lambda functions with Amazon Inspector](#).

Lambda standard scanning + Lambda code scanning

This option combines Lambda standard scanning with Lambda code scanning. When Lambda code scanning is activated Amazon Inspector discovers the Lambda functions and layers in your account and scans for code vulnerabilities your application package dependencies. Lambda code scanning scans the custom application code in your Lambda functions for code vulnerabilities. These two scan types must be activated together. For more information see [Amazon Inspector Lambda code scanning](#).

Activating a scan type

You can activate Amazon Inspector scan types at any time. When you activate a scan type, Amazon Inspector begins scanning eligible resources for the scan type immediately. The following briefly describes each scan type:

[Amazon EC2 scanning](#)

This scan type extracts metadata from your EC2 instance before comparing the metadata against rules collected from security advisories. When you activate this scan type, Amazon Inspector scans all eligible instances in your account for package vulnerabilities and network reachability issues.

[Amazon ECR scanning](#)

This scan type scans container images in Amazon ECR. When you activate this scan type, you change the scanning configuration setting for your private registry from basic scanning to enhanced scanning.

[Lambda standard scanning](#)

Lambda standard scanning is the default Lambda scan type. When you activate Lambda standard scanning, all Lambda functions in your account will be scanned for code vulnerabilities, as long as they were invoked or updated in the last 90 days.

[Lambda code scanning](#)

Lambda code scanning scans custom application code in a Lambda function. When you activate Lambda code scanning, all Lambda functions in your account will be scanned for code vulnerabilities, as long as they were invoked or updated in the last 90 days.

Note

You can either activate Lambda standard scanning or Lambda standard scanning with Lambda code scanning.

For a more comprehensive overview of the available scan types, see [Automated resource scanning with Amazon Inspector](#). This section describes how to activate a scan type in Amazon Inspector.

Activating scans

If you are the delegated administrator for Amazon Inspector in an AWS organization you can enable various Amazon Inspector scan types for multiple accounts in multiple Regions automatically using a shell script developed by Amazon Inspector [inspector2-enablement-with-cli](#) on GitHub. Otherwise, to complete this procedure for a multi-account environment through the console, complete the following steps while signed in as the Amazon Inspector delegated administrator.

Console

To activate scans

1. Open the Amazon Inspector console at <https://console.aws.amazon.com/inspector/v2/home>.
2. Using the AWS Region selector in the upper-right corner of the page, select the Region where you want to activate a new scan type.
3. In the navigation pane, choose **Account management**.
4. On the **Account management** page, select the accounts for which you would like to activate a scan type.
5. Choose **Activate** and select the type of scanning you would like to activate.
6. (Recommended) Repeat these steps in each AWS Region for which you want to activate that scan type.

API

Run the [Enable](#) API operation. In the request, provide the account IDs you are activating scans for, and idempotency token, and one or more of EC2, ECR, LAMBDA, or LAMBDA_CODE for `resourceTypes` to activate scans of that type.

Scanning Amazon EC2 instances with Amazon Inspector

Amazon Inspector Amazon EC2 scanning extracts metadata from your EC2 instance before comparing the metadata against rules collected from security advisories. Amazon Inspector scans instances for package vulnerabilities and network reachability issues to produce [findings](#). Amazon

Inspector performs network reachability scans once every 24 hours and package vulnerability scans on a variable cadence that depends on the scan method associated with the EC2 instance.

Package vulnerability scans can be performed using an [agent-based](#) or [agentless](#) scan method. Both of these scan methods determine how and when Amazon Inspector collects the software inventory from an EC2 instance for package vulnerability scans. Agent-based scanning collects software inventory using the SSM agent, and agentless scanning collects software inventory using on Amazon EBS snapshots.

Amazon Inspector uses the scan methods that you activate for your account. When you activate Amazon Inspector for the first time, your account is automatically enrolled in hybrid scanning, which uses both scan methods. However, you can [change this setting](#) at any time. For information about how to activate a scan type, see [Activating a scan type](#). This section provides information about Amazon EC2 scanning.

Agent-based scanning

Agent-based scans are performed continuously using the SSM agent on all eligible instances. For agent-based scans, Amazon Inspector uses SSM associations, and plugins installed through these associations, to collect software inventory from your instances. In addition to package vulnerability scans for operating system packages, Amazon Inspector agent-based scanning can also detect package vulnerabilities for application programming language packages in Linux-based instances through [Amazon Inspector deep inspection for Linux-based Amazon EC2 instances](#).

The following process explains how Amazon Inspector uses SSM to collect inventory and perform agent-based scans:

1. Amazon Inspector creates SSM associations in your account to collect inventory from your instances. For some Instance types (Windows, and Linux), these associations install plugins on individual instances to collect inventory.
2. Using SSM, Amazon Inspector extracts package inventory from an instance.
3. Amazon Inspector evaluates the extracted inventory and generates findings for any detected vulnerabilities.

Eligible instances

Amazon Inspector will use the agent-based method to scan an instance if it meets the following conditions:

- The instance has a supported OS. For a list of supported OS see the **Agent-based scan support** column of [the section called "Supported operating systems: Amazon EC2 scanning"](#).
- The instance is not excluded from scans by Amazon Inspector EC2 exclusion tags.
- The instance is SSM managed. For instructions on verifying and configuring the agent, see [Configuring the SSM Agent](#).

Agent-based scan behaviors

When using the agent-based scan method, Amazon Inspector initiates new vulnerability scans of EC2 instances in the following situations:

- When you launch a new EC2 instance.
- When you install new software on an existing EC2 instance (Linux and Mac).
- When Amazon Inspector adds a new common vulnerabilities and exposures (CVE) item to its database, and that CVE is relevant to your EC2 instance (Linux and Mac).

Amazon Inspector updates the **Last scanned** field for an EC2 instance when an initial scan is completed. After this, the **Last scanned** field is updated when Amazon Inspector evaluates SSM inventory (every 30 minutes by default), or when an instance is re-scanned because a new CVE impacting that instance was added to the Amazon Inspector database.

You can check when an EC2 instance was last scanned for vulnerabilities from the Instances tab on the **Account management** page, or by using the [ListCoverage](#) command.

Configuring the SSM Agent

In order for Amazon Inspector to detect software vulnerabilities for an Amazon EC2 instance using the agent-based scan method, the instance must be a [managed instance](#) in Amazon EC2 Systems Manager (SSM). An SSM managed instance has the SSM Agent installed and running, and SSM has permission to manage the instance. If you are already using SSM to manage your instances, no other steps are needed for agent-based scans.

The SSM Agent is installed by default on EC2 instances created from some Amazon Machine Images (AMIs). For more information, see [About SSM Agent](#) in the *AWS Systems Manager User Guide*. However, even if it's installed, you may need to activate the SSM Agent manually, and grant SSM permission to manage your instance.

The following procedure describes how to configure an Amazon EC2 instance as a managed instance using an IAM instance profile. The procedure also provides links to more detailed information in the *AWS Systems Manager User Guide*.

[AmazonSSMManagedInstanceCore](#) is the recommended policy to use when you attach an instance profile. This policy has all the permissions needed for Amazon Inspector EC2 scanning.

Note

You can also automate SSM management of all your EC2 instances, without the use of IAM instance profiles using SSM Default Host Management Configuration. For more information, see [Default Host Management Configuration](#).

To configure SSM for an Amazon EC2 instance

1. If it's not already installed by your operating system vendor, install the SSM Agent. For more information, see [Working with SSM Agent](#).
2. Use the AWS CLI to verify that the SSM Agent is running. For more information, see [Checking SSM Agent status and starting the agent](#).
3. Grant permission for SSM to manage your instance. You can grant permission by creating an IAM instance profile and attaching it to your instance. We recommend using the [AmazonSSMManagedInstanceCore](#) policy, because this policy has the permissions for SSM Distributor, SSM Inventory and SSM State manager, that Amazon Inspector needs for scans. For instructions on creating an instance profile with these permissions and attaching it an instance, see [Configure instance permissions for Systems Manager Systems Manager](#).
4. (Optional) Activate automatic updates for the SSM Agent. For more information, see [Automating updates to SSM Agent](#).
5. (Optional) Configure Systems Manager to use an Amazon Virtual Private Cloud (Amazon VPC) endpoint. For more information, see [Create Amazon VPC endpoints](#).

Important

Amazon Inspector requires a Systems Manager State Manager association in your account to collect software application inventory. Amazon Inspector automatically creates an association called `InspectorInventoryCollection-do-not-delete` if one doesn't already exist.

Amazon Inspector also requires a resource data sync and automatically creates one called `InspectorResourceDataSync-do-not-delete` if one doesn't already exist. For more information, see [Configuring resource data sync for Inventory](#) in the *AWS Systems Manager User Guide*. Each account can have a set number of resource data syncs per Region. For more information, see [Maximum number of resource data syncs \(per AWS account per Region\)](#) in [SSM endpoints and quotas](#).

SSM resources created for scanning

Amazon Inspector requires a number of SSM resources in your account to run Amazon EC2 scans. The following resources are created when you first activate Amazon Inspector EC2 scanning:

Note

If any of these SSM resources are deleted while Amazon Inspector Amazon EC2 scanning is activated for your account, Amazon Inspector will attempt to recreate them at the next scan interval.

`InspectorInventoryCollection-do-not-delete`

This is a Systems Manager State Manager (SSM) association that Amazon Inspector uses to collect software application inventory from your Amazon EC2 instances. If your account already has an SSM association for collecting inventory from `InstanceIds*`, Amazon Inspector will use that instead of creating its own.

`InspectorResourceDataSync-do-not-delete`

This is a resource data sync that Amazon Inspector uses to send collected inventory data from your Amazon EC2 instances to an Amazon S3 bucket owned by Amazon Inspector. For more information, see [Configuring resource data sync for Inventory](#) in the *AWS Systems Manager User Guide*.

`InspectorDistributor-do-not-delete`

This is an SSM association Amazon Inspector uses for scanning Windows instances. This association installs the Amazon Inspector SSM plugin on your Windows instances. If the plugin file is inadvertently deleted this association will reinstall it at the next association interval.

InvokeInspectorSsmPlugin-do-not-delete

This is an SSM association Amazon Inspector uses for scanning Windows instances. This association allows Amazon Inspector to initiate scans using the plugin, you can also use it to set custom intervals for scans of Windows instances. For more information, see [Setting custom schedules for Windows instance scans](#).

InspectorLinuxDistributor-do-not-delete

This is an SSM association that Amazon Inspector uses for Amazon EC2 Linux deep inspection. This association installs the Amazon Inspector SSM plugin on your Linux instances.

InvokeInspectorLinuxSsmPlugin-do-not-delete

This is an SSM association Amazon Inspector uses for Amazon EC2 Linux deep inspection. This association allows Amazon Inspector to initiate scans using the plugin.

Note

When you deactivate Amazon Inspector Amazon EC2 scanning or deep inspection, the SSM resource `InvokeInspectorLinuxSsmPlugin-do-not-delete` is no longer invoked.

Agentless scanning

Amazon Inspector uses the agentless scanning method on eligible instances when your account is in hybrid scanning mode. Hybrid scanning mode includes agent-based and agentless scans and is automatically enabled when you activate Amazon EC2 scanning.

For agentless scans, Amazon Inspector uses EBS snapshots to collect a software inventory from your instances. Agentless scanning scans instances for operating system and application programming language package vulnerabilities..

Note

When scanning Linux instances for application programming language package vulnerabilities, the agentless method scans all available paths, whereas agent-based scanning only scans the default paths and additional paths you specify as part of [Amazon Inspector deep inspection for Linux-based Amazon EC2 instances](#). This may result in the

same instance having different findings depending on whether it is scanned using the agent-based method or agentless method.

The following process explains how Amazon Inspector uses EBS snapshots to collect inventory and perform agentless scans:

1. Amazon Inspector creates an EBS snapshot of all volumes attached to the instance. While Amazon Inspector is using it, the snapshot is stored in your account and tagged with `InspectorScan` as a tag key, and a unique scan ID as the tag value.
2. Amazon Inspector retrieves data from the snapshots using [EBS direct APIs](#) and evaluates them for vulnerabilities. Findings are generated for any detected vulnerabilities.
3. Amazon Inspector deletes the EBS snapshots it created in your account.

Eligible instances

Amazon Inspector will use the agentless method to scan an instance if it meets the following conditions:

- The instance has a supported OS. For more information, see the >Agent-based scan support column of [the section called “Supported operating systems: Amazon EC2 scanning”](#).
- The instance has a status of `Unmanaged EC2 instance`, `Stale inventory`, or `No inventory`.
- The instance is backed by Amazon EBS and has one of the following file system formats:
 - `ext3`
 - `ext4`
 - `xf`s
- The instance isn't excluded from scans through Amazon EC2 exclusion tags.
- The number of volumes attached to the instance is less than 8 and have a combined size that's less than or equal to 1200 GB.

Agentless scan behaviors

When your account is configured for **Hybrid scanning**, Amazon Inspector performs agentless scans on eligible instances every 24 hours. Amazon Inspector detects and scans newly eligible

instances every hour, which includes new instances without SSM agents, or pre-existing instances with statuses that have changed to SSM_UNMANAGED.

Amazon Inspector updates the **Last scanned** field for an Amazon EC2 instance whenever it scans extracted snapshots from an instance after an agentless scan.

You can check when an EC2 instance was last scanned for vulnerabilities from the Instances tab on the Account management page, or by using the [ListCoverage](#) command.

Managing scan mode

Your EC2 scan mode determines which scan methods Amazon Inspector will use when performing EC2 scans in your account. You can view the scan mode for your account from the EC2 scanning settings page under **General settings**. Standalone accounts or Amazon Inspector delegated administrators can change the scan mode. When you set the scan mode as the Amazon Inspector delegated administrator that scan mode is set for all member accounts in your organization. Amazon Inspector has the following scan modes:

Agent-based scanning – In this scan mode, Amazon Inspector will exclusively use the agent-based scan method when scanning for package vulnerabilities. This scan mode only scans SSM managed instances in your account, but has the benefit of providing continuous scans in response to new CVE's or changes to the instances. Agent-based scanning also provides Amazon Inspector deep inspection for eligible instances. This is the default scan mode for newly activated accounts.

Hybrid scanning – In this scan mode, Amazon Inspector uses a combination of both agent-based and agentless methods to scan for package vulnerabilities. For eligible EC2 instances that have the SSM agent installed and configured, Amazon Inspector uses the agent-based method. For eligible instances that aren't SSM managed, Amazon Inspector will use the agentless method for eligible EBS-backed instances.

To change the scan mode

1. Sign in using your credentials, and then open the Amazon Inspector console at <https://console.aws.amazon.com/inspector/v2/home>.
2. Using the AWS Region selector in the upper-right corner of the page, select the Region where you want to change your EC2 scan mode.
3. From the side navigation panel, under **General settings**, select **EC2 scanning settings**.
4. Under **Scan Mode**, select **Edit**.
5. Choose a scan mode and then select **Save changes**.

Excluding instances from Amazon Inspector scans

You can exclude Linux and Windows instances from Amazon Inspector scans by tagging these instances with the `InspectorEc2Exclusion` key. Including a tag value is optional. For information about adding tags, see [Tag your Amazon EC2 resources](#).

When you tag an instance for exclusion from Amazon Inspector scans, Amazon Inspector marks the instance as excluded and won't create findings for it. However, the Amazon Inspector SSM plugin will continue to be invoked. To prevent the plugin from being invoked, you must [allow access to tags in instance metadata](#).

Note

You're not charged for excluded instances.

Additionally, you can exclude an encrypted EBS volume from agentless scans by tagging the AWS KMS key used to encrypt that volume with the `InspectorEc2Exclusion` tag. For more information, see [Tagging keys](#).

Supported operating systems

Amazon Inspector scans supported Mac, Windows, and Linux EC2 instance for vulnerabilities in operating system packages. For Linux instances, Amazon Inspector can produce findings for application programming language packages using [Amazon Inspector deep inspection for Linux-based Amazon EC2 instances](#). For Mac and Windows instances only operating system packages are scanned.

For information about supported operating systems, including which operating system can be scanned without an SSM agent, see [Amazon EC2 instances status values](#).

Amazon Inspector deep inspection for Linux-based Amazon EC2 instances

Amazon Inspector expands Amazon EC2 scanning coverage to include deep inspection. With deep inspection, Amazon Inspector detects package vulnerabilities for application programming language packages in your Linux-based Amazon EC2 instances. Amazon Inspector scans default paths for programming language package libraries. However, you can [configure custom paths](#) in addition to the paths that Amazon Inspector scans by default.

Note

You can use deep inspection with the Default Host Management Configuration setting. However, you must create or use a role that's configured with the `ssm:PutInventory` and `ssm:GetParameter` permissions.

To perform deep inspection scans for your Linux-based Amazon EC2 instances, Amazon Inspector uses data collected with the Amazon Inspector SSM plugin. To manage the Amazon Inspector SSM plugin and perform deep inspection for Linux, Amazon Inspector automatically creates the SSM association `InvokeInspectorLinuxSsmPlugin-do-not-delete` in your account. Amazon Inspector collects updated application inventory from your Linux-based Amazon EC2 instances every 6 hours.

Note

Deep inspection is not supported for Windows or Mac instances.

This section describes how to manage Amazon Inspector deep inspection for Amazon EC2 instances, including how to set custom paths for Amazon Inspector to scan.

Topics

- [Accessing or deactivating deep inspection](#)
- [About the Amazon Inspector SSM plugin for Linux](#)
- [Custom paths for Amazon Inspector deep inspection](#)
- [Custom schedules for Amazon Inspector deep inspection](#)
- [Supported programming languages](#)

Accessing or deactivating deep inspection**Note**

For accounts that activate Amazon Inspector after April 17, 2023, deep inspection is automatically activated as part of Amazon EC2 scanning.

To manage deep inspection

1. Sign in using your credentials, and then open the Amazon Inspector console at <https://console.aws.amazon.com/inspector/v2/home>
2. From the navigation pane, choose **General settings**, and then choose Amazon EC2 scanning settings.
3. Under **Deep inspection of Amazon EC2 instance**, you can [set custom paths for your organization or for your own account](#).

You can check the activation status programmatically for a single account with the [GetEc2DeepInspectionConfiguration](#) API. You can check the activation status programmatically for multiple accounts with the [BatchGetMemberEc2DeepInspectionStatus](#) API.

If you activated Amazon Inspector before April 17, 2023, you can activate deep inspection through the console banner or the [UpdateEc2DeepInspectionConfiguration](#) API. If you're the delegated administrator for an organization in Amazon Inspector, you can use the [BatchUpdateMemberEc2DeepInspectionStatus](#) API to activate deep inspection for yourself and your member accounts.

You can deactivate deep inspection through the [UpdateEc2DeepInspectionConfiguration](#) API. Member accounts in an organization can't deactivate deep inspection. Instead, the member account must be deactivated by their delegated administrator using the [BatchUpdateMemberEc2DeepInspectionStatus](#) API.

About the Amazon Inspector SSM plugin for Linux

Amazon Inspector uses the Amazon Inspector SSM plugin to perform deep inspection on your Linux instances. The Amazon Inspector SSM plugin is automatically installed on your Linux instances in the `/opt/aws/inspector/bin` directory. The name of the executable is `inspectorssmplugin`.

Amazon Inspector uses Systems Manager Distributor to deploy the plugin on your instance. To perform deep inspection scans, Systems Manager Distributor and Amazon Inspector must support your Amazon EC2 instance operating system. For information about operating systems that Systems Manager Distributor supports, see [Supported package platforms and architectures](#) in the *AWS Systems Manager User Guide*.

Amazon Inspector creates the following file directories to manage data collected for deep inspection by the Amazon Inspector SSM plugin:

- `/opt/aws/inspector/var/input`
- `/opt/aws/inspector/var/output` – The `packages.txt` file in this directory stores the full paths to packages that deep inspection discovers. If Amazon Inspector detects the same package multiple times on your instance, the `packages.txt` file lists each location where the package was found.

Amazon Inspector stores logs for the plugin in the `/var/log/amazon/inspector` directory.

Uninstalling the Amazon Inspector SSM plugin

If the `inspectorssmplugin` file is inadvertently deleted, the SSM association `InspectorLinuxDistributor-do-not-delete` will try to reinstall the `inspectorssmplugin` file at the next scan interval.

If you deactivate Amazon EC2 scanning, the plugin will be automatically uninstalled from all Linux hosts.

Custom paths for Amazon Inspector deep inspection

You can set custom paths for Amazon Inspector to scan during deep inspection of your Linux Amazon EC2 instances. When you set a custom path, Amazon Inspector scans packages in that directory and all of the sub-directories in it.

All accounts can define up to 5 custom paths. The delegated administrator for an organization can define 10 custom paths.

Amazon Inspector scans all custom paths in addition to the following default paths, which Amazon Inspector scans for all accounts:

- `/usr/lib`
- `/usr/lib64`
- `/usr/local/lib`
- `/usr/local/lib64`

Note

Custom paths must be local paths. Amazon Inspector doesn't scan mapped network paths, such as Network File System mounts or Amazon S3 file system mounts.

Formatting custom paths

A custom path cannot be longer than 256 characters. The following is an example of how a custom path might look:

Example path

```
/home/usr1/project01
```

Note

The package limit per instance is 5,000. The maximum package inventory collection time is 15 minutes. Amazon Inspector recommends that you choose custom paths to avoid these limits.

Setting a custom path in the Amazon Inspector console and with the Amazon Inspector API

The following procedures describe how to set a custom path for Amazon Inspector deep inspection in the Amazon Inspector console and with the Amazon Inspector API. After you set a custom path, Amazon Inspector includes the path in the next deep inspection.

Console

1. Sign in to the AWS Management Console as the delegated administrator, and open the Amazon Inspector console at <https://console.aws.amazon.com/inspector/v2/home>
2. Use the AWS Region selector to choose the Region where you want to activate Lambda standard scanning.
3. From the navigation pane, choose **General settings**, and then choose **EC2 scanning settings**.
4. Under **Custom paths for your own account**, choose **Edit**.
5. In the path text boxes, enter your custom paths.
6. Choose **Save**.

API

Run the [UpdateEc2DeepInspectionConfiguration](#) command. For `packagePaths` specify an array of paths to scan.

Custom schedules for Amazon Inspector deep inspection

By default, Amazon Inspector collects an application inventory from Amazon EC2 instances every 6 hours. However, you can run the following commands to control how often Amazon Inspector does this.

Example command 1: List associations to view association ID and current interval

The following command shows the association ID for the association `InvokeInspectorLinuxSsmPlugin-do-not-delete`.

```
aws ssm list-associations \
--association-filter-list "key=AssociationName,value=InvokeInspectorLinuxSsmPlugin-do-not-delete" \
--region your-Region
```

Example command 2: Update association to include new interval

The following command uses the association ID for the association `InvokeInspectorLinuxSsmPlugin-do-not-delete`. You can set the rate for schedule-expression from 6 hours to a new interval, such as 12 hours.

```
aws ssm update-association \
--association-id "your-association-ID" \
--association-name "InvokeInspectorLinuxSsmPlugin-do-not-delete" \
--schedule-expression "rate(6 hours)" \
--region your-Region
```

Note

Depending on your use case, if you set the rate for schedule-expression from 6 hours to an interval like 30 minutes, you can [exceed the daily ssm inventory limit](#). This causes results to be delayed, and you might encounter Amazon EC2 instances with partial error statuses.

Supported programming languages

For Linux instances, Amazon Inspector deep inspection can produce findings for application programming language packages and operating system packages.

For Mac and Windows instances, Amazon Inspector deep inspection can produce findings only for operating system packages.

For more information about supported programming languages, see [Supported programming languages: Amazon EC2 deep inspection](#).

Scanning Windows EC2 instances with Amazon Inspector

Amazon Inspector automatically discovers all supported Windows instances and includes them in continuous scanning without any extra actions. For information about which instances are supported, see [Operating systems and programming languages supported by Amazon Inspector](#). Amazon Inspector runs Windows scans at regular intervals. Windows instances are scanned at discovery and then every 6 hours. However, you can [adjust the default scan interval](#) after the first scan.

When Amazon EC2 scanning is activated, Amazon Inspector creates the following SSM associations for your Windows resources: `InspectorDistributor-do-not-delete`, `InspectorInventoryCollection-do-not-delete`, and `InvokeInspectorSsmPlugin-do-not-delete`. To install the Amazon Inspector SSM plugin on your Windows instances, the `InspectorDistributor-do-not-delete` SSM association uses the [AWS-ConfigureAWSPackage SSM document](#) and the [AmazonInspector2-InspectorSsmPlugin SSM Distributor package](#). For more information, see [About the Amazon Inspector SSM plugin for Windows](#). To collect instance data and generate Amazon Inspector findings, the `InvokeInspectorSsmPlugin-do-not-delete` SSM association runs the Amazon Inspector SSM plugin at 6-hour intervals. However, you can [customize this setting using a cron or rate expression](#).

Note

Amazon Inspector stages updated Open Vulnerability and Assessment Language (OVAL) definition files to the S3 bucket `inspector2-oval-prod-your-AWS-Region`. The Amazon S3 bucket contains OVAL definitions used in scans. These OVAL definitions shouldn't be modified. Otherwise, Amazon Inspector won't scan for new CVEs when they release.

Amazon Inspector scan requirements for Windows instances

To scan a Windows instance, Amazon Inspector requires the instance to meet the following criteria:

- The instance is an SSM managed instance. For instructions about setting up your instance for scanning, see [Configuring the SSM Agent](#).
- The instance operating system is one of the supported Windows operating systems. For a complete list of supported operating systems, see [Amazon EC2 instances status values](#).
- The instance has the Amazon Inspector SSM plugin installed. Amazon Inspector automatically installs the Amazon Inspector SSM plugin for managed instances upon discovery. See the next topic for details about the plugin.

Note

If your host is running in an Amazon VPC without outgoing internet access, Windows scanning requires your host to be able to access Regional Amazon S3 endpoints. To learn how to configure an Amazon S3 Amazon VPC endpoint, see [Create a gateway endpoint](#) in the *Amazon Virtual Private Cloud User Guide*. If your Amazon VPC endpoint policy is restricting access to external S3 buckets, you must specifically allow access to the bucket maintained by Amazon Inspector in your AWS Region that stores the OVAL definitions used to evaluate your instance. This bucket has the following the format: `inspector2-oval-prod-REGION`.

About the Amazon Inspector SSM plugin for Windows

The Amazon Inspector SSM plugin is required for Amazon Inspector to scan your Windows instances. The Amazon Inspector SSM plugin is automatically installed on your Windows instances in `C:\Program Files\Amazon\Inspector`, and the executable binary file is named `InspectorSsmPlugin.exe`.

The following file locations are created to store data the Amazon Inspector SSM plugin collects:

- `C:\ProgramData\Amazon\Inspector\Input`
- `C:\ProgramData\Amazon\Inspector\Output`
- `C:\ProgramData\Amazon\Inspector\Logs`

By default, the Amazon Inspector SSM plugin runs at below normal priority.

Note

You can use Windows instances with the [Default Host Management Configuration setting](#). However, you must create or use a role that's configured with the `ssm:PutInventory` and `ssm:GetParameter` permissions.

Uninstalling the Amazon Inspector SSM plugin

If the `InspectorSsmPlugin.exe` file is inadvertently deleted, the `InspectorDistributor-do-not-delete` SSM association will reinstall the plugin at the next Windows scan interval. If you want to uninstall the Amazon Inspector SSM plugin, you can use the **Uninstall** action on the `AmazonInspector2-ConfigureInspectorSsmPlugin` document.

Additionally, the Amazon Inspector SSM plugin will be automatically uninstalled from all Windows hosts if you deactivate Amazon EC2 scanning.

Note

If you uninstall the SSM Agent before deactivating Amazon Inspector, the Amazon Inspector SSM plugin will remain on the Windows host but will no longer send data to the Amazon Inspector SSM plugin. For more information, see [Deactivating Amazon Inspector](#).

Setting custom schedules for Windows instance scans

You can customize the time between your Windows Amazon EC2 instance scans by setting a cron expression or rate expression for the `InvokeInspectorSsmPlugin-do-not-delete` association using SSM. For more information, see [Reference: Cron and rate expressions for Systems Manager](#) in the *AWS Systems Manager User Guide* or use the following instructions.

Select from the following code examples to change the scan cadence for Windows instances from the default 6 hours to 12 hours using either a rate expression or a cron expression.

The following examples require you to use the **AssociationId** for the association named `InvokeInspectorSsmPlugin-do-not-delete`. You can retrieve your **AssociationId** by running the following AWS CLI command:

```
$ aws ssm list-associations --association-filter-list  
"key=AssociationName,value=InvokeInspectorSsmPlugin-do-not-delete" --region us-east-1
```

Note

The **AssociationId** is Regional, so you need to first retrieve a unique ID for each AWS Region. You can then run the command to change the scan cadence in each Region where you want to set a custom scan schedule for Windows instances.

Example rate expression

```
$ aws ssm update-association \  
--association-id "YourAssociationId" \  
--association-name "InvokeInspectorSsmPlugin-do-not-delete" \  
--schedule-expression "rate(12 hours)"
```

Example cron expression

```
$ aws ssm update-association \  
--association-id "YourAssociationId" \  
--association-name "InvokeInspectorSsmPlugin-do-not-delete" \  
--schedule-expression "cron(0 0/12 * * ? *)"
```

Scanning Amazon Elastic Container Registry container images with Amazon Inspector

Amazon Inspector scans container images stored in Amazon Elastic Container Registry for software vulnerabilities to generate [package vulnerability findings](#). When you activate Amazon ECR scanning, you set Amazon Inspector as the preferred scanning service for your private registry.

With basic scanning, you can configure your repositories to scan on push or perform manual scans. With enhanced scanning, you scan for operating system and programming language packages vulnerabilities at the registry level. For a side-by-side comparison of the differences between basic and enhanced scanning, see the [Amazon Inspector FAQ](#).

Note

Basic scanning is provided and billed through Amazon ECR. For more information, see [Amazon Elastic Container Registry pricing](#). Enhanced scanning is provided and billed through Amazon Inspector. For more information, see [Amazon Inspector pricing](#).

For information about how to activate Amazon ECR scanning, see [Activating a scan type](#). For information about how to view your findings, see [Managing findings in Amazon Inspector](#). For information about how to view your findings at the image level, see [Image scanning](#) in the *Amazon Elastic Container Registry User Guide*. You can also manage findings in AWS services not available for basic scanning, like [AWS Security Hub and Amazon EventBridge](#).

This section provides information about Amazon ECR scanning and describes how to configure enhanced scanning for Amazon ECR repositories.

Scan behaviors for Amazon ECR scanning

When you first activate ECR scanning, and your repository is configured for continuous scanning, Amazon Inspector detects all eligible images that you have pushed within 30 days, or pulled within the last 90 days. Then Amazon Inspector scans the detected images and sets their scan status to active. Amazon Inspector continues to monitor images as long as they were pushed or pulled within the last 90 days (by default), or within the ECR rescan duration you configure. For more information, see [Configuring the Amazon ECR re-scan duration](#).

For continuous scanning, Amazon Inspector initiates new vulnerability scans of container images in the following situations:

- Whenever a new container image is pushed.
- Whenever Amazon Inspector adds a new common vulnerabilities and exposures (CVE) item to its database, and that CVE is relevant to that container image (continuous scanning only).

If you configure your repository for on push scanning, images are only scanned when you push them.

You can check when a container image was last checked for vulnerabilities from the **Container images** tab on the **Account management** page, or by using the [ListCoverage](#) API. Amazon

Inspector updates the **Last scanned at** field of an Amazon ECR image in response to the following events:

- When Amazon Inspector completes an initial scan of a container image.
- When Amazon Inspector re-scans a container image because a new common vulnerabilities and exposures (CVE) item that impacts that container image was added to the Amazon Inspector database.

Supported operating systems and media types

For information about supported operating systems, see [Supported operating systems: Amazon ECR scanning with Amazon Inspector](#).

Amazon Inspector scans of Amazon ECR repositories cover the following supported media types:

- "application/vnd.docker.distribution.manifest.v1+json"
- "application/vnd.docker.distribution.manifest.v1+prettyjws"
- "application/vnd.oci.image.manifest.v1+json"
- "application/vnd.docker.distribution.manifest.v2+json"

Note

Scratch images and "application/vnd.docker.distribution.manifest.list.v2+json" images aren't supported.

Configuring the Amazon ECR re-scan duration

The Amazon ECR re-scan duration setting determines how long Amazon Inspector continuously monitors container images in repositories. You configure the re-scan duration for the image push date and image pull date. As a best practice, configure the re-scan duration to best suit your environment. For example, if you build images often, choose a shorter scan duration. For images used over long periods of time, choose a longer scan duration. The default scan duration for new accounts, including new accounts added to an organization, is 90 days. Amazon Inspector will continue to monitor and rescan an image as long as it's been pushed or pulled within the configured push and pull dates. If the image hasn't been pushed or pulled within the configured push and pull dates, Amazon Inspector stops monitoring it. When Amazon Inspector

stops monitoring an image, it sets the image scan status code to `inactive` and reason code to `expired`. Amazon Inspector then schedules all associated image findings to be closed. If you increase the push date duration, Amazon Inspector applies the change to all actively scanned images in repositories configured for continual scanning. However, inactive images remain inactive, even if you pushed them within the new duration.

Note

When you configure the re-scan duration from a delegated administrator account, Amazon Inspector applies the setting to all member accounts in the organization.

Image push date duration

The image push date duration determines how long Amazon Inspector continuously monitors images after they were pushed to repositories following the latest pull date. The following options are available as re-scan durations:

- 14 days
- 30 days
- 60 days
- 90 days (default)
- 180 days
- Lifetime

Image pull date duration

The image pull date duration determines how long Amazon Inspector continuously monitors images after the latest pull date. The following options are available as re-scan durations:

- 14 days
- 30 days
- 60 days
- 90 days (default)
- 180 days

To configure the Amazon ECR re-scan duration

1. Sign in using your credentials, and then open the Amazon Inspector console at <https://console.aws.amazon.com/inspector/v2/home>.
2. Select the AWS Region where you want to configure the Amazon ECR re-scan duration.
3. From the navigation pane, choose **General settings**, and then choose **ECR scanning settings**.
4. On **ECR scanning settings**, under **ECR re-scan duration**, choose the image push date duration and image pull date duration that you want to set.
5. Choose **Save**.

Scanning AWS Lambda functions with Amazon Inspector

Amazon Inspector support for AWS Lambda functions and layers provides continuous automated security vulnerability assessments. Amazon Inspector offers two types of Lambda function scanning:

[Amazon Inspector Lambda standard scanning](#)

This is the default Lambda scan type. Lambda standard scanning scans application dependencies within a Lambda function and layers for [package vulnerabilities](#).

[Amazon Inspector Lambda code scanning](#)

This scan type scans the custom application code in your Lambda function and layers for [code vulnerabilities](#). You can either activate Lambda standard scanning or activate Lambda standard scanning with Lambda code scanning..

When you activate Lambda function scanning, Amazon Inspector creates the following [AWS CloudTrail service-linked channels](#) in your account: `cloudtrail:CreateServiceLinkedChannel` and `cloudtrail>DeleteServiceLinkedChannel`. Amazon Inspector manages these channels and uses them to monitor your CloudTrail events for scans. These channels allow you to see CloudTrail events in your account as if you had a trail in CloudTrail. We recommend you create your own trail in CloudTrail to manage events for your account.

For information about how to activate Lambda function scanning, see [Activating a scan type](#). This section provides information about Lambda function scanning.

Scan behaviors for Lambda function scanning

Upon activation, Amazon Inspector scans all Lambda functions invoked or updated in the last 90 days in your account. Amazon Inspector initiates vulnerability scans of Lambda functions in the following situations:

- As soon as Amazon Inspector discovers an existing Lambda function.
- When you deploy a new Lambda function to the Lambda service.
- When you deploy an update to the application code or dependencies of an existing Lambda function or its layers.
- Whenever Amazon Inspector adds a new common vulnerabilities and exposures (CVE) item to its database, and that CVE is relevant to your function.

Amazon Inspector monitors each Lambda function throughout its lifetime until it's either deleted or excluded from scanning.

You can check when a Lambda function was last checked for vulnerabilities from the **Lambda functions** tab on the **Account management** page, or by using the [ListCoverage](#) API. Amazon Inspector updates the **Last scanned at** field for a Lambda function in response to the following events:

- When Amazon Inspector completes an initial scan of a Lambda function.
- When a Lambda function is updated.
- When Amazon Inspector re-scans a Lambda function because a new CVE item impacting that function was added to the Amazon Inspector database.

Supported runtimes and eligible functions

Amazon Inspector supports different runtimes for Lambda standard scanning and Lambda code scanning. For a list of supported runtimes for each scan type, see [Supported runtimes: Amazon Inspector Lambda standard scanning](#) and [Supported runtimes: Amazon Inspector Lambda code scanning](#).

In addition to having a supported runtime, a Lambda function needs to meet the following criteria to be eligible for Amazon Inspector scans:

- The function has been invoked or updated in the last 90 days.

- The function is marked \$LATEST.
- The function isn't excluded from scans by tags.

Note

Lambda functions that haven't been invoked or modified in the last 90 days are automatically excluded from scans. Amazon Inspector will resume scanning an automatically excluded function if it is invoked again or if changes are made to the Lambda function code.

Amazon Inspector Lambda standard scanning

Amazon Inspector Lambda standard scanning identifies software vulnerabilities in the application package dependencies you add to your Lambda function code and layers. For example, if your Lambda function uses a version of the `python-jwt` package with a known vulnerability, Lambda standard scanning will generate a finding for that function.

If Amazon Inspector detects a vulnerability in your Lambda function application package dependencies, Amazon Inspector produces a detailed **Package Vulnerability** type finding.

For instructions on activating a scan type see [Activating a scan type](#).

Note

Lambda standard scanning doesn't scan the AWS SDK dependency installed by default in the Lambda runtime environment. Amazon Inspector only scans dependencies uploaded with the function code or inherited from a layer.

Note

Deactivating Amazon Inspector Lambda standard scanning will also deactivate Amazon Inspector Lambda code scanning.

Excluding functions from Lambda standard scanning

You can add tags to Lambda functions, so you can exclude them from Amazon Inspector Lambda standard scans. Excluding functions from scans can prevent unactionable alerts. When you tag a function for exclusion, the tag must have the following key-value pair.

- Key:InspectorExclusion
- Value:LambdaStandardScanning

This topic describes how to tag a function for exclusion from scans. For more information about adding tags in Lambda, see [Using tags on Lambda functions](#).

To exclude a function from scans

1. Sign in using your credentials, and then open the Lambda console at <https://console.aws.amazon.com/lambda/>.
2. From the navigation pane, choose **Functions**.
3. Choose the name of the function you would want to exclude from Amazon Inspector Lambda standard scans.
4. Choose **Configuration**, and then choose **Tags**.
5. Choose **Manage tags**, and then **Add new tag**.
 - a. For **Key**, enter InspectorExclusion.
 - b. For **Value**, enter LambdaStandardScanning
6. Choose **Save**.

Amazon Inspector Lambda code scanning

Important

This feature captures snippets of Lambda functions to highlight detected vulnerabilities. These snippets can show hardcoded credentials and other sensitive materials.

With this feature, Amazon Inspector scans application code in a Lambda function for code vulnerabilities based on AWS security best practices to detect data leaks, injection flaws, missing

encryption, and weak cryptography. Amazon Inspector uses automated reasoning and machine learning to evaluate your Lambda function application code. It also uses internal detectors that are developed in collaboration with Amazon CodeGuru to identify policy violations and vulnerabilities. For more information, see the [CodeGuru Detector Library](#).

Amazon Inspector generates a [code vulnerability](#) when it detects a vulnerability in your Lambda function application code. This finding type includes a code snippet showing the issue and where you can find the issue in your code. It also suggests how to remediate the issue. The suggestion includes plug-and-play code blocks that you can use to replace vulnerable lines of code. These code fixes are provided in addition to general code remediation guidance for this finding type.

Code remediation suggestions are powered by automated reasoning and generative artificial intelligence services. Some code remediation suggestions might not work as intended. You are responsible for the code remediation suggestions you adopt. Always review code remediation suggestions before adopting them. You might need to edit them to make sure your code performs as intended. For more information, see the [Responsible AI Policy](#).

Lambda code scanning can be activated by itself or together with Lambda standard scanning. For more information, see [Activating a scan type](#). For information about which AWS Regions support this feature, see [Region-specific feature availability](#).

Encrypting your code in code vulnerability findings

CodeGuru stores code snippets that are detected to be in connection with a code vulnerability finding using Lambda code scanning. By default, CodeGuru controls [the AWS owned key](#) used to encrypt your code. However, you can use your own customer managed key for encryption through the Amazon Inspector API. For more information, see [Encryption at rest for code in your findings](#)

Excluding functions from Lambda code scanning

You can add tags to Lambda functions, so you can exclude them from Amazon Inspector Lambda code scans. Excluding functions from scans can prevent unactionable alerts. When you tag a function for exclusion, the tag must have the following key-value pair.

- Key – `InspectorCodeExclusion`
- Value – `LambdaCodeScanning`

This topic describes how to tag a function for exclusion from code scans. For more information about adding tags in Lambda, see [Using tags on Lambda functions](#).

To exclude a function from code scans

1. Sign in using your credentials, and then open the Lambda console at <https://console.aws.amazon.com/lambda/>.
2. From the navigation pane, choose **Functions**.
3. Choose the name of the function you would want to exclude from Amazon Inspector Lambda code scans.
4. Choose **Configuration**, and then choose **Tags**.
5. Choose **Manage tags**, and then **Add new tag**.
 - a. For **Key**, enter `InspectorExclusion`.
 - b. For **Value**, enter `LambdaCodeScanning`
6. Choose **Save**.

Deactivating a scan type in Amazon Inspector

This section describes how to deactivate a scan type. When you deactivate a scan type, you lose access to any findings the scan type produced. If you [reactivate the scan type](#), Amazon Inspector scans all eligible resources to generate new findings.

Tip

If you want to keep a record of your findings, you can export them to an Amazon Simple Storage Service (Amazon S3) bucket as a findings report. For more information, see [Exporting Amazon Inspector findings reports](#).

When you deactivate a scan type, you might encounter the following changes in the AWS account where you deactivated the scan type:

[Amazon EC2 scanning](#)

When you deactivate Amazon Inspector Amazon EC2 scanning for an account, the following SSM associations are deleted:

- `InspectorDistributor-do-not-delete`
- `InspectorInventoryCollection-do-not-delete`

- InspectorLinuxDistributor-do-not-delete
- InvokeInspectorLinuxSsmPlugin-do-not-delete
- InvokeInspectorSsmPlugin-do-not-delete.

Additionally, the Amazon Inspector SSM plugin installed through this association is removed from all of your Windows hosts. For more information, see [Scanning Windows EC2 instance](#).

[Amazon ECR scanning](#)

When you deactivate Amazon ECR scanning for an account, the Amazon ECR scan type account changes from **Enhanced scanning** with Amazon Inspector to **Basic scanning** with Amazon ECR.

[Lambda standard scanning](#)

When you deactivate Lambda standard scanning for an account, you deactivate Lambda code scanning if the scan type was activated. You also delete the CloudTrail service-linked channel that Amazon Inspector created when you activated Lambda standard scanning.

Deactivating scans

Deactivating all scan types for an account deactivates Amazon Inspector for that account in that AWS Region. For more information, see [Deactivating Amazon Inspector](#).

To complete this procedure for a multi-account environment, follow these steps while signed in as the Amazon Inspector delegated administrator.

Console

To deactivate scans

1. Sign in using your credentials, and then open the Amazon Inspector console at <https://console.aws.amazon.com/inspector/v2/home>.
2. By using the AWS Region selector in the upper-right corner of the page, select the Region where you want to deactivate scans.
3. In the navigation pane, choose **Account management**.
4. Choose the **Accounts** tab to show the scanning status of an account.
5. Select the check box of each account for which you want to deactivate scans.

6. Choose **Actions**, and, from the **Deactivate** options, select the scan type you wish to deactivate.
7. (Recommended) Repeat these steps in each AWS Region for which you want to deactivate that scan type.

API

Run the [Disable](#) API operation. In the request, provide the account IDs you are deactivating scans for, and for `resourceTypes` provide one or more of EC2, ECR, LAMBDA, or LAMBDA_CODE to deactivate scans.

Center for Internet Security (CIS) scans for Amazon EC2 instance operating systems

Amazon Inspector CIS scans (CIS scans) benchmark your Amazon EC2 instance operating systems to make sure you configured them according to best practice recommendations established by the Center for Internet Security. [CIS Security Benchmarks](#) provides industry standard configuration baselines and best practices for configuring a system securely. You can perform or schedule CIS scans after you enable Amazon Inspector EC2 scanning for an account. For information about how to activate Amazon EC2 scanning, see [Activating a scan type](#).

Amazon Inspector performs CIS scans on target Amazon EC2 instances based on instance tags and your defined scanning schedule. Amazon Inspector performs a series of instance checks on each targeted instance. Each check evaluates whether your system configuration meets specific CIS Benchmark recommendations. Each check has a CIS check ID and title, which correspond with a CIS Benchmark recommendation for that platform. When a CIS scan completes, you can view the results to see which instance checks passed, skipped, or failed for that system.

Note

To perform or schedule CIS scans, you must have a secure internet connection. However, if you want to run CIS scans on private instances, you must use a VPC endpoint.

Topics

- [Amazon EC2 instance requirements for Amazon Inspector CIS scans](#)
- [Running CIS scans](#)
- [Considerations for managing Amazon Inspector CIS scans with AWS Organizations](#)
- [Amazon Inspector owned Amazon S3 buckets used for Amazon Inspector CIS scans](#)
- [Creating a CIS scan configuration](#)
- [Viewing CIS scan results](#)
- [Editing a CIS scan configuration](#)
- [Downloading a CIS scan results](#)

Amazon EC2 instance requirements for Amazon Inspector CIS scans

To run a CIS scan on your Amazon EC2 instance, the Amazon EC2 instance must meet the following criteria:

- The instance operating system is one of the supported operating systems for CIS scans. For more information, see [Operating systems and programming languages supported by Amazon Inspector](#).
- The instance is an Amazon EC2 Systems Manager instance. For more information, see [Working with the SSM Agent](#) in the *AWS Systems Manager User Guide*.
- The Amazon Inspector SSM plugin is installed on the instance. Amazon Inspector automatically installs this plugin on managed instances.
- The instance has an instance profile that grants permissions for SSM to manage the instance and Amazon Inspector to run CIS scans for that instance. To grant these permissions, attach the [AmazonSSMManagedInstanceCore](#) and [AmazonInspector2ManagedCisPolicy](#) policies to an IAM role. Then attach the IAM role to your instance as an instance profile. For instructions on creating and attaching an instance profile, see [Work with IAM roles](#) in the *Amazon EC2 User Guide*.

Note

You're not required to enable Amazon Inspector deep inspection before running a CIS scan on your Amazon EC2 instance. If you disable Amazon Inspector deep inspection, Amazon Inspector automatically installs the SSM Agent, but the SSM Agent won't be invoked to run deep inspection anymore. However, as a result, the `InspectorLinuxDistributor-donot-delete` association is present in your account.


Amazon Virtual Private Cloud endpoint requirements for running CIS scans on private Amazon EC2 instances

You can run CIS scans on Amazon EC2 instances over an Amazon network. However, if you want to run CIS scans on private Amazon EC2 instances, you must [create Amazon VPC endpoints](#). The following endpoints are required when you create Amazon VPC endpoints for Systems Manager:

- `amazonaws.com.region.ec2messages`

- `amazonaws.com.region.inspector2`
- `amazonaws.com.region.s3`
- `amazonaws.com.region.ssm`
- `amazonaws.com.region.ssmmessages`

For more information, see [Creating Amazon VPC endpoints for Systems Manager](#) in the *AWS Systems Manager User Guide*.

 **Note**

Currently, some AWS Regions don't support the `amazonaws.com.region.inspector2` endpoint.

Running CIS scans


You can either run a CIS scan once on-demand or as a scheduled recurring scan. To run a scan, you first create a scan configuration.

When you create a scan configuration, you specify tag key-value pairs to use to target instances. If you are the Amazon Inspector delegated administrator for an organization, you can specify multiple accounts in the scan configuration, and Amazon Inspector will look for instances with the specified tags in each of those accounts. You choose the CIS Benchmark level for the scan. For each benchmark, CIS supports a level 1 and level 2 profile designed to provide baselines for different levels of security that different environments may require.

- **Level 1** – recommends essential basic security settings that can be configured on any system. Implementing these settings should cause little or no interruption of service. The goal of these recommendations is to reduce the number of entry points into your systems, reducing your overall cybersecurity risks.
- **Level 2** – recommends more advanced security settings for high-security environments. Implementing these settings requires planning and coordination to minimize the risk of business impact. The goal of these recommendations is to help you achieve regulatory compliance.

Level 2 extends level 1. When you choose Level 2, Amazon Inspector checks for all configurations recommended for level 1 and level 2.

After defining the parameters for your scan, you can choose whether to run it as a one time scan, which runs after you complete the configuration, or a recurring scan. Recurring scans can run daily, weekly, or monthly, at a time of your choice.

 Tip

We recommend choosing a day and time that's least likely to impact your system while the scan is running.

Considerations for managing Amazon Inspector CIS scans with AWS Organizations

When you run CIS scans in an organization, Amazon Inspector delegated administrators and member accounts interact with CIS scan configurations and scan results differently.

How Amazon Inspector delegated administrators can interact with CIS scan configurations and scan results

When the delegated administrator creates a scan configuration, either for all accounts or a specific member accounts, the organization owns the configuration. Scan configurations that an organization owns have an ARN specifying the organization ID as the owner:

```
arn:aws:inspector2:Region:111122223333:owner/OrganizationId/cis-configuration/scanId
```

The delegated administrator can manage scan configurations that an organization owns, even if another account created them.

The delegated administrator can view scan results for any account in its organization.

If the delegated administrator creates a scan configuration and specifies SELF as the target account, the delegated administrator owns scan configuration, even if they leave the organization. However, the delegated administrator cannot change the target of a scan configuration with SELF as the target.

Note

The delegated administrator cannot add tags to CIS scan configurations the organization owns.

How Amazon Inspector member accounts can interact with CIS scan configurations and scan results

When a member account creates a CIS scan configuration, it owns the configuration. However, the delegated administrator can view the configuration. If a member account leaves the organization, the delegated administrator won't be able to view the configuration.

Note

The delegated administrator cannot edit a scan configuration the member account creates.

Member accounts, delegated administrators with SELF as the target, and standalone accounts all own scan configurations they create. These scan configurations have an ARN that shows the account ID as the owner:

```
arn:aws:inspector2:Region:111122223333:owner/111122223333/cis-configuration/scanId
```

A member account can view scan results in their account, including scan results from CIS scans the delegated administrator scheduled.

Amazon Inspector owned Amazon S3 buckets used for Amazon Inspector CIS scans

Open Vulnerability and Assessment Language (OVAL) is an information security effort that standardizes how to assess and report the machine state of computer systems. The following table lists all of the Amazon Inspector owned Amazon S3 buckets with OVAL definitions that are used for CIS scans. Amazon Inspector stages OVAL definition files that are required for CIS scans. The Amazon Inspector owned Amazon S3 buckets should be allowlisted in VPCs if necessary.

Note

The details for each of the following Amazon Inspector owned Amazon S3 buckets aren't subject to change. However, the table might be updated to reflect newly supported AWS Regions. You can't use Amazon Inspector owned Amazon S3 buckets for other Amazon S3 operations or in your own Amazon S3 buckets.

CIS bucket	AWS Region
<code>cis-datasets-prod-arn-5908f6f</code>	Europe (Stockholm)
<code>cis-datasets-prod-bah-8f88801</code>	Middle East (Bahrain)
<code>cis-datasets-prod-bjs-0f40506</code>	China (Beijing)
<code>cis-datasets-prod-bom-435a167</code>	Asia Pacific (Mumbai)
<code>cis-datasets-prod-cdg-f3a9c58</code>	Europe (Paris)
<code>cis-datasets-prod-cgk-09eb12f</code>	Asia Pacific (Jakarta)
<code>cis-datasets-prod-cmh-63030b9</code>	US East (Ohio)
<code>cis-datasets-prod-cpt-02c5c6f</code>	Africa (Cape Town)
<code>cis-datasets-prod-dub-984936f</code>	Europe (Ireland)
<code>cis-datasets-prod-fra-6eb96eb</code>	Europe (Frankfurt)
<code>cis-datasets-prod-gru-de69f99</code>	South America (São Paulo)
<code>cis-datasets-prod-hkg-8e30800</code>	Asia Pacific (Hong Kong)
<code>cis-datasets-prod-iad-8438411</code>	US East (N. Virginia)
<code>cis-datasets-prod-icn-f4eff1c</code>	Asia Pacific (Seoul)
<code>cis-datasets-prod-kix-5743b21</code>	Asia Pacific (Osaka)

CIS bucket	AWS Region
cis-datasets-prod-lhr-8b1fbd0	Europe (London)
cis-datasets-prod-mxp-7b1bbce	Europe (Milan)
cis-datasets-prod-nrt-464f684	Asia Pacific (Tokyo)
cis-datasets-prod-osu-5bead6f	AWS GovCloud (US-East)
cis-datasets-prod-pdt-adadf9c	AWS GovCloud (US-West)
cis-datasets-prod-pdx-acfb052	US West (Oregon)
cis-datasets-prod-sfo-1515ba8	US West (N. California)
cis-datasets-prod-sin-309725b	Asia Pacific (Singapore)
cis-datasets-prod-syd-f349107	Asia Pacific (Sydney)
cis-datasets-prod-yul-5e0c95e	Canada (Central)
cis-datasets-prod-zhy-5a8eacb	China (Ningxia)
cis-datasets-prod-zrh-67e0e3d	Europe (Zurich)

Creating a CIS scan configuration

This topic describes how to create a CIS scan configuration.

To run a CIS scan

1. Sign in using your credentials, and then open the Amazon Inspector console at <https://console.aws.amazon.com/inspector/v2/home>.
2. Use the AWS Region dropdown to select the AWS Region where you want to run a CIS scan.
3. From the navigation pane, choose **On-demand scans**, and then choose **CIS scans**.
4. Choose **Create new scan**.
5. For **Scan configuration name**, enter a **Scan configuration name**.

6. For **Target resource tags**, enter a **Key** and corresponding **Value** for the instances you want to scan. You can specify up to five different values for each key and a total of 25 tags to include in the scan.
7. For **CIS Benchmark level**, you can select **Level 1** for basic security configurations or **Level 2** for advanced security configurations.
8. For **Target accounts**, specify which accounts to include in the CIS scan. For more information, see [Considerations for managing Amazon Inspector CIS scans with AWS Organizations](#).

If your account is the delegated administrator account, you can select **All accounts** or **Specify accounts**. The **All accounts** option targets all accounts in your organization. The **Specify accounts** only targets individual accounts in your organization. If you choose this option, you can specify more than one account by separating the account numbers with a comma. You can also enter SELF instead of an account ID to create a scan configuration for your account

If your account is a standalone account or member account in an organization, you can select **Self** to create a scan configuration for your account.

9. For **Schedule**, choose **One time scan**, which runs as soon as you finish creating your scan configuration, or **Recurring scans**, which runs at the time you specify.
10. Confirm your choices, and then choose **Create**.

Viewing CIS scan results

Amazon Inspector creates a scan job for every scan configuration that runs and collects results of a scan with a unique scan ID. CIS scan results are available for 90 days. You can view CIS scan results by its checks or scanned resources:

- **Scan results aggregated by checks** – Groups the results of a scan by each individual check performed during the scan. For each check, you get a report of how many resources failed, skipped, or passed.
- **Scan results aggregated by scanned resources** – Groups the results of a scan by each scanned resource the scan targets during the scan. For each resource, you get a report of how many checks that a resource failed, skipped, or passed.

This topic describes how to view results for a CIS scan.

To view scan results

1. Sign in using your credentials, and then open the Amazon Inspector console at <https://console.aws.amazon.com/inspector/v2/home>.
2. Use the AWS Region dropdown to select the AWS Region where you created your CIS scan configuration.
3. From the navigation pane, choose **On-demand scans**, and then choose **CIS scans**.
4. Choose the **Scan results** tab.
5. Under the **Scheduled by** column, choose the scan schedule ID you want to view. Or select the row with the scan schedule ID you want to view, and then choose **View details**.
6. Choose **Checks** to view each check that was performed or **Scanned resources** to view each scanned resource that was targeted during the scan.

You can also view details for scheduled CIS scans.

To view details for scheduled CIS scans

1. Sign in using your credentials, and then open the Amazon Inspector console at <https://console.aws.amazon.com/inspector/v2/home>.
2. Use the AWS Region dropdown to select the AWS Region where you created your CIS scan configuration.
3. From the navigation pane, choose **On-demand scans**, and then choose **CIS scans**.
4. Choose the **Scheduled** tab.
5. Under the **Scan configuration name** column, choose the name of the scan configuration you want to view. Or select the row with the scan configuration you want to view, and then choose **View details**.

Editing a CIS scan configuration

This topic describes how to edit a CIS scan configuration.

To edit a CIS scan configuration

1. Sign in using your credentials, and then open the Amazon Inspector console at <https://console.aws.amazon.com/inspector/v2/home>.

2. Use the AWS Region dropdown to select the AWS Region where you created your CIS scan configuration.
3. From the navigation pane, choose **On-demand scans**, and then choose **CIS scans**.
4. Choose the **Scheduled** tab.
5. Select the row with the scan configuration you want to edit, and then choose **Edit**.

Downloading a CIS scan results

You can download a PDF or CSV of a CIS scan using the Amazon Inspector console or API.

Note

You can only download a CSV file of your CIS scan results for CIS scans collected after 05/03/2024.

This topic describes how to download a CIS scan using the Amazon Inspector console.

To download CIS scan results from the console

1. Sign in using your credentials, and then open the Amazon Inspector console at <https://console.aws.amazon.com/inspector/v2/home>.
2. Use the AWS Region dropdown to select the AWS Region where you created your CIS scan configuration.
3. From the navigation pane, choose **On-demand scans**, and then choose **CIS scans**.
4. Choose the **Scan results** tab.
5. Under the **Scheduled By** column, choose the scan schedule ID that you want to view. Or select the row with the scan schedule ID that you want to view, and then choose **View details**.
6. Choose **Download**, and then choose **PDF** or **CSV**. If your account is the delegated administrator account, you can choose **Select account** to download results for a specific member account.

Understanding Amazon Inspector findings

Amazon Inspector generates a finding when it detects a vulnerability in an Amazon EC2 instance, a container image in Amazon ECR, or an AWS Lambda function. A finding is a detailed report about a vulnerability impacting one of your AWS resources.

Findings are named after vulnerabilities and provide severity ratings, information about impacted AWS resources, and details that describe how to remediate detected vulnerabilities. Amazon Inspector stores all of your active findings until you remediate them.

When you remediate a finding, Amazon Inspector automatically closes the finding. After seven days, Amazon Inspector automatically deletes the finding.

Note

Amazon Inspector will reopen a closed finding within seven days of closing the finding if the issue that caused the vulnerability reoccurs.

If you disable Amazon Inspector, findings are removed after 24 hours. If a resource is terminated, any finding related to the resource is removed after seven days. If AWS suspends your account, findings are removed after 90 days. Findings for stopped instances remain active.

Findings states

Amazon Inspector categorizes findings in the following states.

Active

Amazon Inspector categorizes a finding that hasn't been remediated as **Active**.

Suppressed

Amazon Inspector categorizes a finding subject to one or more [suppression rules](#) as **Suppressed**.

Closed

When a finding has been remediated, Amazon Inspector categorizes the finding as **Closed**.

Topics

- [Amazon Inspector finding types](#)
- [Viewing your Amazon Inspector findings](#)
- [Viewing details for your Amazon Inspector findings](#)
- [Viewing the Amazon Inspector score and understanding vulnerability intelligence details](#)
- [Understanding severity levels for your Amazon Inspector findings](#)

Amazon Inspector finding types

This section describes the different finding types in Amazon Inspector.

Topics

- [Package vulnerability](#)
- [Code vulnerability](#)
- [Network reachability](#)

Package vulnerability

Package vulnerability findings identify software packages in your AWS environment that are exposed to Common Vulnerabilities and Exposures (CVEs). Attackers can exploit these unpatched vulnerabilities to compromise the confidentiality, integrity, or availability of data, or to access other systems. The CVE system is a reference method for publicly known information security vulnerabilities and exposures. For more information, see <https://www.cve.org/>.

Amazon Inspector can generate package vulnerability findings for EC2 instances, ECR container images, and Lambda functions. Package vulnerability findings have additional details unique to this finding type, these are the [Inspector score and vulnerability intelligence](#).

Code vulnerability

Code vulnerability findings identify lines in your code that attackers could exploit. Code vulnerabilities include injection flaws, data leaks, weak cryptography, or missing encryption in your code.

Amazon Inspector evaluates your Lambda function application code using automated reasoning and machine learning that analyzes your application code for overall security compliance. It identifies policy violations and vulnerabilities based on internal detectors developed in

collaboration with Amazon CodeGuru. For a list of possible detections, see [CodeGuru Detector Library](#).

Important

Amazon Inspector code scanning captures code snippets to highlight detected vulnerabilities. These snippets may show hardcoded credentials or other sensitive materials in plaintext.

Amazon Inspector can generate **code vulnerability** findings for Lambda functions if you enable [Amazon Inspector Lambda code scanning](#).

Code snippets detected in connection with a code vulnerability are stored by the CodeGuru service. By default an [AWS owned key](#) controlled by CodeGuru is used to encrypt your code, however, you can use your own customer managed key for encryption through the Amazon Inspector API. For more information see [Encryption at rest for code in your findings](#).

Network reachability

Network reachability findings indicate that there are open network paths to Amazon EC2 instances in your environment. These findings appear when your TCP and UDP ports are reachable from the VPC edges, such as an internet gateway (including instances behind Application Load Balancers or Classic Load Balancers), a VPC peering connection, or a VPN through a virtual gateway. These findings highlight network configurations that may be overly permissive, such as mismanaged security groups, Access Control Lists, or internet gateways, or that may allow for potentially malicious access.

Amazon Inspector only generates network reachability findings for Amazon EC2 instances. Amazon Inspector performs scans for network reachability findings every 24 hours once Amazon Inspector is enabled.

Amazon Inspector evaluates the following configurations when scanning for network paths:

- [Amazon EC2 instances](#)
- [AWS Lambda functions](#)
- [Application Load Balancers](#)
- [Direct Connect](#)

- [Elastic Load Balancers](#)
- [Elastic Network Interfaces](#)
- [Internet Gateways](#)
- [Network Access Control Lists](#)
- [Route Tables](#)
- [Security Groups](#)
- [Subnets](#)
- [Virtual Private Clouds](#)
- [Virtual Private Gateways](#)
- [VPC endpoints](#)
- [VPC gateway endpoints](#)
- [VPC peering connections](#)
- [VPN connections](#)

Viewing your Amazon Inspector findings

You can view your Amazon Inspector findings in the Amazon Inspector console and with the Amazon Inspector [ListFindings](#) API. In the Amazon Inspector console, you can view your findings in the Amazon Inspector dashboard and on the **Findings** screen. You can also view your findings in [AWS Security Hub and Amazon Elastic Container Registry \(Amazon ECR\)](#). By default, the Amazon Inspector dashboard and **Findings** screen show your active findings. You can also view your findings by category. The procedures in this section describe how to view your findings in Amazon Inspector console and with the Amazon Inspector API.


Console

To view Amazon Inspector findings

1. Sign in using your credentials, and then open the Amazon Inspector console at <https://console.aws.amazon.com/inspector/v2/home>.
2. (Optional) From the navigation pane, choose **Dashboard**. The dashboard shows an overview of the coverage for your environment and only your critical findings.
3. (Optional) From the navigation pane, choose **Findings**. The **Findings** screen shows all of your active findings in a table where you can [filter your findings](#) by status and filter criteria.


You can also create [suppression rules](#) to exclude findings from view. You can view details for a finding by choosing the name of the finding.

4. (Optional) From the navigation pane, choose one of the following options to view your findings by category:
 - By vulnerability – Shows your most critical vulnerabilities.
 - By account – Shows all of your accounts and the scan coverage and total number of findings with [critical and high severity ratings](#).

 **Note**

This category is only available to delegated administrators.

- By instance – Shows your most vulnerable Amazon EC2 instances.

 **Note**

The findings grouped in this category don't include information about network availability.

- By container image – Shows your most vulnerable Amazon ECR container images.
- By container repository – Shows your most vulnerable repositories.
- By Lambda function – Shows your most vulnerable Lambda functions.

API

To view Amazon Inspector findings

- Run the [ListFindings](#) API operation. In the request, specify [filterCriteria](#) to return specific findings.

Viewing details for your Amazon Inspector findings

The procedure in this section describes how to view details for Amazon Inspector findings.

To view the details for a finding

1. Sign in using your credentials, and then open the Amazon Inspector console at <https://console.aws.amazon.com/inspector/v2/home>
2. Select the Region to view findings in.
3. In the navigation pane, choose **Findings** to display the findings list
4. (Optional) Use the filter bar to select a specific finding. For more information, see [Filtering your Amazon Inspector findings](#).
5. Choose a finding to view its details panel.

The **Finding details** panel contains the basic identifying features of the finding. This includes the title of the finding as well as a basic description of the vulnerability identified, remediation suggestions, and a severity score. For information about scoring, see [Understanding severity levels for your Amazon Inspector findings](#).

The details available for a finding varies depending on finding type and the **Resource affected**.

All findings contain the AWS account ID number the finding was identified for, a severity, a finding **Type**, the date the finding was created at, and a **Resource affected** section with details about that resource.

The finding **Type** determines the remediation and vulnerability intelligence information available for the finding. Depending on the finding type, different finding details are available.

Package Vulnerability


Package vulnerability findings are available for EC2 instances, ECR container images, and Lambda functions. See [Package vulnerability](#) for more info.

Package vulnerability findings also include [Viewing the Amazon Inspector score and understanding vulnerability intelligence details](#).

This finding type has the following details:

- **Fix available** – Indicates if the vulnerability is fixed in a newer version of the affected packages. Has one of the following values:
 - YES, which means all the affected packages have a fixed version.
 - NO, which means no affected packages have a fixed version.

- **PARTIAL**, which means one or more (but not all) of the affected packages have a fixed version.
- **Exploit available** – Indicates the vulnerability has a known exploit.
 - **YES**, which means the vulnerability discovered in your environment has a known exploit. Amazon Inspector doesn't have visibility into the use of exploits in an environment.
 - **NO**, which means this vulnerability doesn't have a known exploit.
- **Affected packages** – Lists each package identified as vulnerable in the finding, and the details of each package:
- **Filepath** – The EBS volume ID and partition number associated with a finding. This field is present in findings for EC2 instances scanned using [Agentless scanning](#).
- **Installed version / Fixed version** – The version number of the currently installed package that a vulnerability was detected for. Compare the installed version number with the value after the slash (/). The second value is the version number of the package that fixes the detected vulnerability as provided by the Common Vulnerabilities and Exposures (CVEs) or advisory associated with the finding. If the vulnerability has been fixed in multiple versions, this field lists the most recent version that includes the fix. If a fix isn't available, this value is `None available`.

 **Note**

If a finding was detected before Amazon Inspector began including this field in findings, the value for this field is empty. However, a fix may be available.

- **Package manager** – The package manager used to configure this package.
- **Remediation** – If a fix is available through an updated package or programming library, this section includes the commands that you can run to make the update. You can copy the provided command and run it in your environment.

 **Note**

Remediation commands are provided from vendor data feeds and may vary depending on your system configuration. Review finding references or operating system documentation for more specific guidance.

- **Vulnerability details** – provides a link to the Amazon Inspector preferred source for the CVE identified in the finding, such as National Vulnerability Database (NVD), REDHAT, or another

OS vendor. Additionally, you will find the severity scores for the finding. For more information about severity scoring such as, see [Understanding severity levels for your Amazon Inspector findings](#). The following scores are included, including the scoring vectors for each:

- [Exploit Prediction Scoring System \(EPSS\) score](#)
- Inspector score
- CVSS 3.1 from Amazon CVE
- CVSS 3.1 from NVD
- CVSS 2.0 from NVD (where applicable, for older CVEs)
- **Related vulnerabilities** – Specifies other vulnerabilities related to the finding. Typically these are other CVEs that impact the same package version, or other CVEs within the same group as the finding CVE, as determined by the vendor.

Code vulnerability

Code vulnerability findings are available for Lambda functions only. See [Code vulnerability](#) for more info. This finding type has the following details:

- **Fix available** – For code vulnerabilities this value is always YES.
- **Detector name** – The name of the CodeGuru detector used to detect the code vulnerability. For a list of possible detections, see the [CodeGuru Detector Library](#).
- **Detector tags** – The CodeGuru tags associated with the detector, CodeGuru uses tags to categorize detections.
- **Relevant CWE** – IDs of the Common Weakness Enumeration (CWE)s associated with the code vulnerability.
- **File path** – The file location of the code vulnerability.
- **Vulnerability location** – For Lambda code scanning code vulnerabilities, this field shows the exact lines of code where Amazon Inspector found the vulnerability.
- **Suggested remediation** – This suggests how the code can be edited to remediate the finding.

Network reachability

Network reachability findings are only available for EC2 instances. See [Network reachability](#) for more info. This finding type has the following details:

- **Open port range** – The port range through which the EC2 instance could be accessed.
- **Open network paths** – Shows the open access path to the EC2 instance. Select an item on the path for more information.
- **Remediation** – Recommends a method for closing the open network path.

Viewing the Amazon Inspector score and understanding vulnerability intelligence details

Amazon Inspector creates a score for Amazon Elastic Compute Cloud (Amazon EC2) instance findings. You can view the Amazon Inspector score and vulnerability intelligence details in the Amazon Inspector console. The Amazon Inspector score provides you with details that you can compare with metrics in the [Common Vulnerability Scoring System](#). These details are only available for [package vulnerability](#) findings. This section describes how to interpret the Amazon Inspector score and understand vulnerability intelligence details.

Amazon Inspector score

The Amazon Inspector score is a contextualized score that Amazon Inspector creates for each EC2 instance finding. The Amazon Inspector score is determined by correlating the base CVSS v3.1 score information with information collected from your compute environment during scans, such as network reachability results and exploitability data. For example, the Amazon Inspector score of a finding may be lower than the base score if the vulnerability is exploitable over the network but Amazon Inspector determines that no open network path to the vulnerable instance is available from the internet.

The base score for a finding is the CVSS v3.1 base score provided by the vendor. RHEL, Debian, or Amazon vendor base scores are supported, for other vendors, or cases where the vendor hasn't provided a score Amazon Inspector uses the base score from the [National Vulnerability Database](#) (NVD). Amazon Inspector uses the [Common Vulnerability Scoring System Version 3.1 Calculator](#) to calculate the score. You can see the source of the base score of an individual finding in the finding's details under **vulnerability details**, as **Vulnerability source** (or `packageVulnerabilityDetails.source` in the finding JSON)

Note

Amazon Inspector score isn't available for Linux instances running Ubuntu. This is because Ubuntu defines its own vulnerability severity that may differ from the associated CVE severity.

Amazon Inspector score details

When you open the details page of a finding you can select the **Inspector score and vulnerability intelligence** Tab. This panel shows the difference between the base score and the **Inspector score**. This section explains how Amazon Inspector assigned the severity rating based on a combination of the Amazon Inspector score and the vendor score for the software package. If the scores differ this panel shows an explanation of why.

In the **CVSS score metrics** section you can see a table with comparisons between the CVSS base score metrics and the **Inspector score**. The metrics compared are the base metrics defined in the [CVSS specification document](#) maintained by first.org. The following is a summary of the base metrics:

Attack Vector

The context by which a vulnerability can be exploited. For Amazon Inspector findings this can be Network, **Adjacent Network**, or **Local**.

Attack Complexity

This describes the level of difficulty an attacker will face when exploiting the vulnerability. A **Low** score means that the attacker will need to meet little or no additional conditions to exploit the vulnerability. A **High** score means that an attacker will need invest a considerable amount of effort in order carry out a successful attack with this vulnerability.

Privilege Required

This describes the level of privilege an attacker will need to exploit a vulnerability.

User Interaction

This metric states if a successful attack using this vulnerability requires a human user, other than the attacker.

Scope

This states whether a vulnerability in one vulnerable component impacts resources in components beyond the vulnerable component's security scope. If this value is **Unchanged** the affected resource and the impacted resource are the same. If this value is **Changed** then the vulnerable component can be exploited to impact resources managed by different security authorities.

Confidentiality

This measures the level of impact to the confidentiality of data within a resource when the vulnerability is exploited. This ranges from **None**, where no confidentiality is lost, to **High** where all information within a resource is divulged or confidential information such as passwords or encryption keys can be divulged.

Integrity

This measures the level of impact to the integrity of data within the impacted resource if the vulnerability is exploited. Integrity is at risk when the attacker to modify files within impacted resources. The score ranges from **None**, where the exploit does not allow an attacker to modify any information, to **High**, where if exploited, the vulnerability would allow an attacker to modify any or all files, or the files that could be modified have serious consequences.

Availability

This measures the level of impact to the availability of the impacted resource when the vulnerability is exploited. The score ranges from **None**, when the vulnerability does not impact availability at all, to **High**, where if exploited, the attacker can completely deny availability to the resource, or cause a service to become unavailable.

Vulnerability Intelligence

This section summarizes available intelligence about the CVE from Amazon as well as industry standard security intelligence sources such as Recorded Future, and Cybersecurity and Infrastructure Security Agency (CISA).

Note

Intel from CISA, Amazon, or Recorded Future won't be available for all CVEs.

You can view vulnerability intelligence details in the console or by using the [BatchGetFindingDetails](#) API. The following details are available in the console:

ATT&CK

This section shows the MITRE tactics, techniques, and procedures (TTPs) associated with the CVE. The associated TTPs are shown, if there are more than two applicable TTPs you can select

the link to see a complete list. Selecting a tactic or technique opens information about it on the MITRE website.

CISA

This section covers relevant dates associated with the vulnerability. The date Cybersecurity and Infrastructure Security Agency (CISA) added the vulnerability to Known Exploited Vulnerabilities Catalog, based on evidence of active exploitation, and the Due date CISA expects systems to be patched by. This information is sourced from CISA.

Known malware

This section lists known exploit kits and tools that exploit this vulnerability.

Evidence

This section summarizes the most critical security events involving this vulnerability. If more than 3 events have the same criticality level the top three most recent events are displayed.

Last time reported

This section shows the Last known public exploit date for this vulnerability.

Understanding severity levels for your Amazon Inspector findings

When Amazon Inspector generates a finding, it assigns a severity rating to the finding. Severity ratings help you assess and prioritize your findings. The severity rating for a finding corresponds to a numerical score and level: **informational**, **low**, **medium**, **high**, and **critical**. Amazon Inspector determines the severity rating for a finding based on the [finding type](#). This section describes how Amazon Inspector determines a severity rating for each finding type.

Software package vulnerability severity

Amazon Inspector uses the NVD/CVSS score as the basis of severity scoring for software package vulnerabilities. The NVD/CVSS score is the vulnerability severity score published by the NVD and defined by the CVSS. The NVD/CVSS score is a composition of security metrics, such as attack complexity, exploit code maturity, and privileges required. Amazon Inspector produces a numerical score from 1 to 10 that reflects the vulnerability's severity. Amazon Inspector categorizes this as a base score because it reflects the severity of a vulnerability according to its intrinsic characteristics, which are constant over time. This score also assumes the reasonable worst-case impact across

different deployed environments. [The CVSS v3 standard](#) maps CVSS scores to the following severity ratings.

Score	Rating
0	Informational
0.1–3.9	Low
4.0–6.9	Medium
7.0–8.9	High
9.0–10.0	Critical

Package vulnerability findings can also have a severity of **Untriaged**. This means that the vendor hasn't yet set a vulnerability score for the detected vulnerability. In this case, we recommend using the reference URLs for the finding to research that vulnerability and respond accordingly.

Package vulnerability findings include the following scores and associated scoring vectors as part of their finding details:

- EPSS score
- Inspector score
- CVSS 3.1 from Amazon CVE
- CVSS 3.1 from NVD
- CVSS 2.0 from NVD (where applicable)

Code vulnerability severity

For code vulnerability findings Amazon Inspector uses the severity levels defined by the Amazon CodeGuru detectors that generated the finding. Each detector is assigned a severity using the CVSS v3 scoring system. For an explanation of the severities CodeGuru uses see [Severity definitions](#) in the CodeGuru guide. For a list of detectors by severity, select from the supported programming languages below:

- [Python detectors by severity](#)

- [Java detectors by severity](#)

Network reachability severity

Amazon Inspector determines the severity for a network reachability vulnerability based on the service, ports, and protocols that are exposed and by the type of open path. The following table defines these severity ratings. The value in the **Open path rating** column represents open paths from virtual gateways, peered VPCs, and AWS Direct Connect networks. All other exposed services, ports, and protocols have an Informational severity rating.

Service	TCP ports	UDP ports	Internet path rating	Open path rating
DHCP	67, 68, 546, 547	67, 68, 546, 547	Medium	Informational
Elasticsearch	9300, 9200	NA	Medium	Informational
FTP	21	21	High	Medium
Global catalog LDAP	3268	NA	Medium	Informational
Global catalog LDAP over TLS	3269	NA	Medium	Informational
HTTP	80	80	Low	Informational
HTTPS	443	443	Low	Informational
Kerberos	88, 464, 543, 544, 749, 751	88, 464, 749, 750, 751, 752	Medium	Informational
LDAP	389	389	Medium	Informational
LDAP over TLS	636	NA	Medium	Informational
MongoDB	27017, 27018, 27019, 28017	NA	Medium	Informational
MySQL	3306	NA	Medium	Informational

NetBIOS	137, 139	137, 138	Medium	Informational
NFS	111, 2049, 4045, 1110	111, 2049, 4045, 1110	Medium	Informational
Oracle	1521, 1630	NA	Medium	Informational
PostgreSQL	5432	NA	Medium	Informational
Print services	515	NA	High	Medium
RDP	3389	3389	Medium	Low
RPC	111, 135, 530	111, 135, 530	Medium	Informational
SMB	445	445	Medium	Informational
SSH	22	22	Medium	Low
SQL Server	1433	1434	Medium	Informational
Syslog	601	514	Medium	Informational
Telnet	23	23	High	Medium
WINS	1512, 42	1512, 42	Medium	Informational

Managing findings in Amazon Inspector

With Amazon Inspector, you can manage your findings in different ways. You can filter your findings based on their status. You can search your findings based on filter criteria. You can create suppression rules to exclude findings from your list of findings. You can also export findings to AWS Security Hub, Amazon EventBridge, and Amazon Simple Storage Service (Amazon S3).

Topics

- [Filtering your Amazon Inspector findings](#)
- [Suppressing Amazon Inspector findings](#)
- [Exporting Amazon Inspector findings reports](#)
- [Creating custom responses to Amazon Inspector findings with Amazon EventBridge](#)

Filtering your Amazon Inspector findings

You can filter your Amazon Inspector findings using filter criteria. If a finding doesn't match your filter criteria, Amazon Inspector excludes the finding from view. This section describes how to filter your Amazon Inspector findings using filter criteria.

Creating filters in the Amazon Inspector console

In each findings view, you can use the filter functionality to locate findings with specific characteristics. Filters are removed when you move to a different tabbed view.

A filter is made up of a filter criteria, which consists of a filter attribute paired with a filter value. Findings that do not match your filter criteria are excluded from the findings list. For example, to see all findings that are associated with your administrator account, you can choose the AWS account ID attribute and pair it with the value of your twelve digit AWS account ID.

Some filter criteria apply to all findings, while others are available for specific resource types or finding types only.

To apply a filter to the findings view

1. Sign in using your credentials, and then open the Amazon Inspector console at <https://console.aws.amazon.com/inspector/v2/home>.

2. In the navigation pane, choose **Findings**. The default view displays all findings with an **Active** status.
3. To filter findings by criteria, select the *Add filter* bar to see a list of all applicable filter criteria for that view. Different filter criteria are available in different views.
4. Choose a criterion that you want to filter by from the list.
5. From the criterion input pane enter the desired filter values to define that criterion.
6. Choose **Apply** to apply that filter criterion to your current results. You can continue to add other filter criterion by selecting the filter input bar again.
7. (Optional) To view your suppressed or closed findings, choose **Active** in the filter bar, and then choose **Suppressed** or **Closed**. Choose **Show all** to see active, suppressed, and closed findings in the same view.

Suppressing Amazon Inspector findings

You can create suppression rules to hide findings that match criteria. For example, you can create a suppression rule to hide findings based on their severity ratings. If Amazon Inspector generates a finding that matches your suppression rule, Amazon Inspector suppresses the finding and hides it from view. Amazon Inspector stores suppressed findings until they're remediated. Once a suppressed finding is remediated, Amazon Inspector closes the finding. You can view suppressed findings in the console.

You create suppression rules to prioritize your most important findings. Suppression rules don't have any impact on your findings, as they only hide findings from view. You cannot create a suppression rule that closes or remediates findings. You can also [suppress unwanted findings in AWS Security Hub with an Amazon EventBridge rule](#). The procedures in this section describe how to create, view, edit, and delete a suppression rule.

Note

Only the delegated administrator for an organization can create and manage suppression rules.

Creating a suppression rule

You can create suppression rules to filter the list of findings that are shown by default. You can create a suppression rule programmatically by using the [CreateFilter](#) API and specifying SUPPRESS as the value for `action`.

Note

Only stand alone accounts and Amazon Inspector delegated administrators can create and manage suppression rules. Members in an organization will not see an option for suppression rules in the navigation pane.

To create a suppression rule (console)

1. Sign in using your credentials, and then open the Amazon Inspector console at <https://console.aws.amazon.com/inspector/v2/home>.
2. In the navigation pane, choose **Suppression rules**. Then choose **Create rule**.
3. For each criterion, do the following:
 - Select the filter bar to see a list of filter criteria that you can add to your suppression rule.
 - Select the filter criteria for your suppression rule.
4. When you have finished adding criteria, enter a name for the rule and an optional description.
5. Choose **Save rule**. Amazon Inspector immediately applies the new suppression rule and hides any findings that match the criteria.

Viewing suppressed findings

By default, Amazon Inspector does not display suppressed findings in the Amazon Inspector console. However, you can view the findings suppressed by a particular rule.

To view suppressed findings

1. Sign in using your credentials, and then open the Amazon Inspector console at <https://console.aws.amazon.com/inspector/v2/home>.
2. In the navigation pane, select **Suppression rules**.
3. In the suppression rules list, select the title of the rule.

Changing suppression rules

You can make changes to suppression rules at any time.

To modify suppression rules

1. Sign in using your credentials, and then open the Amazon Inspector console at <https://console.aws.amazon.com/inspector/v2/home>.
2. In the navigation pane, select **Suppression rules**.
3. Select the title of the suppression rule that you want to modify.
4. Make the intended changes, then choose **Save** to update the rule.

Deleting suppression rules

You can delete suppression rules. If you delete a suppression rule, Amazon Inspector stops suppressing new and existing occurrences of findings that meet the rule criteria and that aren't suppressed by other rules.

After you delete a suppression rule, new and existing occurrences of findings that met the rule's criteria have a status of **Active**. This means that they appear by default on the Amazon Inspector console. In addition, Amazon Inspector publishes these findings to AWS Security Hub and Amazon EventBridge as events.

To delete a suppression rule

1. Sign in using your credentials, and then open the Amazon Inspector console at <https://console.aws.amazon.com/inspector/v2/home>.
2. In the navigation pane, select **Suppression rules**.
3. Select the check box next to the title of the suppression rule you want to delete.
4. Choose **Delete**, and then confirm your choice to permanently delete the rule.

Exporting Amazon Inspector findings reports

A findings report is a CSV or JSON file that provides a detailed snapshot of your findings. You can export a findings report to AWS Security Hub, Amazon EventBridge, and Amazon Simple Storage Service (Amazon S3). When you configure a findings report, you specify which findings to include

in it. By default, your findings report includes data for all of your active findings. If you're the delegated administrator for an organization, your findings report includes data for all member accounts in the organization. To customize a findings report, create and apply [a filter](#) to it.

When you export a findings report, Amazon Inspector encrypts your findings data with an AWS KMS key that you specify. After Amazon Inspector encrypts your findings data, it stores your finding report in an Amazon S3 bucket that you specify. Your AWS KMS key must be used in the same AWS Region as your Amazon S3 bucket. Your AWS KMS key policy must allow Amazon Inspector to use it, and your Amazon S3 bucket policy must allow Amazon Inspector to add objects to it. After you export your findings report, you can download it from your Amazon S3 bucket or transfer it to new location. You can also use your Amazon S3 bucket as a repository for other exported findings reports.

This section describes how to export a findings report in the Amazon Inspector console. The following tasks require that you verify your permissions, configure an Amazon S3 bucket, configure an AWS KMS key, and configure and export a findings report.

Note

If you export a findings report with the Amazon Inspector [CreateFindingsReport](#) API, you can only view your active findings. If you want to view your suppressed or closed findings, you must specify SUPPRESSED or CLOSED as part of your [filter criteria](#).

Tasks

- [Step 1: Verify your permissions](#)
- [Step 2: Configure an S3 bucket](#)
- [Step 3: Configure an AWS KMS key](#)
- [Step 4: Configure and export a findings report](#)
- [Troubleshoot export errors](#)

Step 1: Verify your permissions

Note

After you export a findings report for the first time, steps 1–3 are optional. Following these steps is based on whether you want to use the same Amazon S3 bucket and AWS

KMS key for other exported findings reports. If you want to export a findings report programmatically after completing steps 1–3, use the [CreateFindingsReport](#) operation of the Amazon Inspector API.

Before you export a findings report from Amazon Inspector, verify that you have the permissions that you need to both export findings reports and configure resources for encrypting and storing the reports. To verify your permissions, use AWS Identity and Access Management (IAM) to review the IAM policies that are attached to your IAM identity. Then compare the information in those policies to the following list of actions that you must be allowed to perform to export a findings report.

Amazon Inspector

For Amazon Inspector, verify that you're allowed to perform the following actions:

- `inspector2:ListFindings`
- `inspector2:CreateFindingsReport`

These actions allow you to retrieve findings data for your account and to export that data in findings reports.

If you plan to export large reports programmatically, you might also verify that you're allowed to perform the following actions: `inspector2:GetFindingsReportStatus`, to check the status of reports, and `inspector2:CancelFindingsReport`, to cancel exports that are in progress.

AWS KMS

For AWS KMS, verify that you're allowed to perform the following actions:

- `kms:GetKeyPolicy`
- `kms:PutKeyPolicy`

These actions allow you to retrieve and update the key policy for the AWS KMS key that you want Amazon Inspector to use to encrypt your report.

To use the Amazon Inspector console to export a report, also verify that you're allowed to perform the following AWS KMS actions:

- `kms:DescribeKey`
- `kms:ListAliases`

These actions allow you to retrieve and display information about the AWS KMS keys for your account. You can then choose one of these keys to encrypt your report.

If you plan to create a new KMS key for encryption of your report, you also need to be allowed to perform the `kms:CreateKey` action.

Amazon S3

For Amazon S3, verify that you're allowed to perform the following actions:

- `s3:CreateBucket`
- `s3:DeleteObject`
- `s3:PutBucketAcl`
- `s3:PutBucketPolicy`
- `s3:PutBucketPublicAccessBlock`
- `s3:PutObject`
- `s3:PutObjectAcl`

These actions allow you to create and configure the S3 bucket where you want Amazon Inspector to store your report. They also allow you to add and delete objects from the bucket.

If you plan to use the Amazon Inspector console to export your report, also verify that you're allowed to perform the `s3:ListAllMyBuckets` and `s3:GetBucketLocation` actions. These actions allow you to retrieve and display information about the S3 buckets for your account. You can then choose one of these buckets to store the report.

If you're not allowed to perform one or more of the required actions, ask your AWS administrator for assistance before you proceed to the next step.

Step 2: Configure an S3 bucket

After you verify your permissions, you're ready to configure the S3 bucket where you want to store your findings report. It can be an existing bucket for your own account, or an existing bucket that's owned by another AWS account and you're allowed to access. If you want to store your report in a new bucket, create the bucket before you proceed.

The S3 bucket must be in the same AWS Region as the findings data that you want to export. For example, if you're using Amazon Inspector in the US East (N. Virginia) Region and you want to export findings data for that Region, the bucket must also be in the US East (N. Virginia) Region.

In addition, the bucket's policy must allow Amazon Inspector to add objects to the bucket. This topic explains how to update the bucket policy and it provides an example of the statement to add to the policy. For detailed information about adding and updating bucket policies, see [Using bucket policies](#) in the *Amazon Simple Storage Service User Guide*.

If you want to store your report in an S3 bucket that's owned by another account, work with the bucket's owner to update the bucket's policy. Also obtain the URI for the bucket. You'll need to enter this URI when you export your report.

To update the bucket policy

1. Sign in using your credentials, and then open the Amazon S3 console at <https://console.aws.amazon.com/s3>.
2. In the navigation pane, choose **Buckets**.
3. Choose the S3 bucket where you want to store the findings report.
4. Choose the **Permissions** tab.
5. In the **Bucket policy** section, choose **Edit**.
6. Copy the following example statement to your clipboard:

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "allow-inspector",
      "Effect": "Allow",
      "Principal": {
        "Service": "inspector2.amazonaws.com"
      },
      "Action": [
        "s3:PutObject",
        "s3:PutObjectAcl",
        "s3:AbortMultipartUpload"
      ],
      "Resource": "arn:aws:s3:::amzn-s3-demo-bucket/*",
      "Condition": {
        "StringEquals": {
          "aws:SourceAccount": "111122223333"
        }
      },
      "ArnLike": {
        "aws:SourceArn": "arn:aws:inspector2:Region:111122223333:report/*"
      }
    }
  ]
}
```



```
    }  
  }  
}  
]  
}
```

7. In the **Bucket policy** editor on the Amazon S3 console, paste the preceding statement into the policy to add it to the policy.

When you add the statement, ensure that the syntax is valid. Bucket policies use JSON format. This means that you need to add a comma before or after the statement, depending on where you add the statement to the policy. If you add the statement as the last statement, add a comma after the closing brace for the preceding statement. If you add it as the first statement or between two existing statements, add a comma after the closing brace for the statement.

8. Update the statement with the correct values for your environment, where:
 - *amzn-s3-demo-bucket* is the name of the bucket.
 - *111122223333* is the account ID for your AWS account.
 - *Region* is the AWS Region in which you're using Amazon Inspector and want to allow Amazon Inspector to add reports to the bucket. For example, *us-east-1* for the US East (N. Virginia) Region.

Note

If you're using Amazon Inspector in a manually enabled AWS Region, also add the appropriate Region code to the value for the `Service` field. This field specifies the Amazon Inspector service principal.

For example, if you're using Amazon Inspector in the Middle East (Bahrain) Region, which has the Region code `me-south-1`, replace `inspector2.amazonaws.com` with `inspector2.me-south-1.amazonaws.com` in the statement.

Note that the example statement defines conditions that use two IAM global condition keys:

- [aws:SourceAccount](#) – This condition allows Amazon Inspector to add reports to the bucket only for your account. It prevents Amazon Inspector from adding reports to the bucket for

other accounts. More specifically, the condition specifies which account can use the bucket for the resources and actions specified by the `aws:SourceArn` condition.

To store reports for additional accounts in the bucket, add the account ID for each additional account to this condition. For example:

```
"aws:SourceAccount": [111122223333,444455556666,123456789012]
```

- [aws:SourceArn](#) – This condition restricts access to the bucket based on the source of the objects that are being added to the bucket. It prevents other AWS services from adding objects to the bucket. It also prevents Amazon Inspector from adding objects to the bucket while performing other actions for your account. More specifically, the condition allows Amazon Inspector to add objects to the bucket only if the objects are findings reports, and only if those reports are created by the account and in the Region specified in the condition.

To allow Amazon Inspector to perform the specified actions for additional accounts, add Amazon Resource Names (ARNs) for each additional account to this condition. For example:

```
"aws:SourceArn": [  
  "arn:aws:inspector2:Region:111122223333:report/*",  
  "arn:aws:inspector2:Region:444455556666:report/*",  
  "arn:aws:inspector2:Region:123456789012:report/*"  
]
```

The accounts specified by the `aws:SourceAccount` and `aws:SourceArn` conditions should match.

Both conditions help prevent Amazon Inspector from being used as a [confused deputy](#) during transactions with Amazon S3. Although we don't recommend it, you can remove these conditions from the bucket policy.

9. When you finish updating the bucket policy, choose **Save changes**.

Step 3: Configure an AWS KMS key

After you verify your permissions and configure the S3 bucket, determine which AWS KMS key you want Amazon Inspector to use to encrypt your findings report. The key must be a customer managed, symmetric encryption KMS key. In addition, the key must be in the same AWS Region as the S3 bucket that you configured to store the report.

The key can be an existing KMS key from your own account, or an existing KMS key that another account owns. If you want to use a new KMS key, create the key before proceeding. If you want to use an existing key that another account owns, obtain the Amazon Resource Name (ARN) of the key. You'll need to enter this ARN when you export your report from Amazon Inspector. For information about creating and reviewing the settings for KMS keys, see [Managing keys](#) in the *AWS Key Management Service Developer Guide*.

After you determine which KMS key you want to use, give Amazon Inspector permission to use the key. Otherwise, Amazon Inspector won't be able to encrypt and export the report. To give Amazon Inspector permission to use the key, update the key policy for the key. For detailed information about key policies and managing access to KMS keys, see [Key policies in AWS KMS](#) in the *AWS Key Management Service Developer Guide*.

Note

The following procedure is for updating an existing key to allow Amazon Inspector to use it. If you don't have an existing key, see [Creating keys](#) in the *AWS Key Management Service Developer Guide*.

To update the key policy

1. Sign in using your credentials, and then open the AWS KMS console at <https://console.aws.amazon.com/kms>.
2. In the navigation pane, choose **Customer managed keys**.
3. Choose the KMS key that you want to use to encrypt the report. The key must be a symmetric encryption (**SYMMETRIC_DEFAULT**) key.
4. On the **Key policy** tab, choose **Edit**. If you do not see a key policy with an **Edit** button, you must first select **Switch to policy view**.
5. Copy the following example statement to your clipboard:

```
{
  "Sid": "Allow Amazon Inspector to use the key",
  "Effect": "Allow",
  "Principal": {
    "Service": "inspector2.amazonaws.com"
  },
  "Action": [
```

```
    "kms:Decrypt",
    "kms:GenerateDataKey*"
  ],
  "Resource": "*",
  "Condition": {
    "StringEquals": {
      "aws:SourceAccount": "111122223333"
    },
    "ArnLike": {
      "aws:SourceArn": "arn:aws:inspector2:Region:111122223333:report/*"
    }
  }
}
```

6. In the **Key policy** editor on the AWS KMS console, paste the preceding statement into the key policy to add it to the policy.

When you add the statement, ensure that the syntax is valid. Key policies use JSON format. This means that you need to add a comma before or after the statement, depending on where you add the statement to the policy. If you add the statement as the last statement, add a comma after the closing brace for the preceding statement. If you add it as the first statement or between two existing statements, add a comma after the closing brace for the statement.

7. Update the statement with the correct values for your environment, where:
 - **111122223333** is the account ID for your AWS account.
 - **Region** is the AWS Region in which you want to allow Amazon Inspector to encrypt reports with the key. For example, `us-east-1` for the US East (N. Virginia) Region.

Note

If you're using Amazon Inspector in a manually enabled AWS Region, also add the appropriate Region code to the value for the `Service` field. For example, if you're using Amazon Inspector in the Middle East (Bahrain) Region, replace `inspector2.amazonaws.com` with `inspector2.me-south-1.amazonaws.com`.

Like the example statement for the bucket policy in the preceding step, the `Condition` fields in this example use two IAM global condition keys:

- [aws:SourceAccount](#) – This condition allows Amazon Inspector to perform the specified actions only for your account. More specifically, it determines which account can perform the specified actions for the resources and actions specified by the `aws:SourceArn` condition.

To allow Amazon Inspector to perform the specified actions for additional accounts, add the account ID for each additional account to this condition. For example:

```
"aws:SourceAccount": [111122223333,444455556666,123456789012]
```

- [aws:SourceArn](#) – This condition prevents other AWS services from performing the specified actions. It also prevents Amazon Inspector from using the key while performing other actions for your account. In other words, it allows Amazon Inspector to encrypt S3 objects with the key only if the objects are findings reports, and only if those reports are created by the account and in the Region specified in the condition.

To allow Amazon Inspector to perform the specified actions for additional accounts, add ARNs for each additional account to this condition. For example:

```
"aws:SourceArn": [  
  "arn:aws:inspector2:us-east-1:111122223333:report/*",  
  "arn:aws:inspector2:us-east-1:444455556666:report/*",  
  "arn:aws:inspector2:us-east-1:123456789012:report/*"  
]
```

The accounts specified by the `aws:SourceAccount` and `aws:SourceArn` conditions should match.

These conditions help prevent Amazon Inspector from being used as a [confused deputy](#) during transactions with AWS KMS. Although we don't recommend it, you can remove these conditions from the statement.

8. When you finish updating the key policy, choose **Save changes**.

Step 4: Configure and export a findings report

Note

You only can export only one findings report a time. If an export is currently in progress, you must wait until the export is complete before exporting another findings report.

After you verify your permissions and you configure resources to encrypt and store your findings report, you're ready to configure and export the report.

To configure and export a findings report

1. Sign in using your credentials, and then open the Amazon Inspector console at <https://console.aws.amazon.com/inspector/v2/home>.
2. In the navigation pane, under **Findings**, choose **All findings**.
3. (Optional) By using the filter bar above the **Findings** table, [add filter criteria](#) that specify which findings to include in the report. As you add criteria, Amazon Inspector updates the table to include only those findings that match the criteria. The table provides a preview of the data that your report will contain.

Note

We recommend that you add filter criteria. If you don't, the report will include data for all of your findings in the current AWS Region that have a status of **Active**. If you're the Amazon Inspector administrator for an organization, this includes findings data for all the member accounts in your organization.

If a report includes data for all or many findings, it can take a long time to generate and export the report, and you can export only one report at a time.

4. Choose **Export findings**.
5. In the **Export settings** section, for **Export file type**, specify a file format for the report:
 - To create a JavaScript Object Notation (.json) file that contains the data, choose **JSON**.
If you choose the **JSON** option, the report will include all the fields for each finding. For a list of possible JSON fields see the [Finding](#) data type in the Amazon Inspector API reference.
 - To create a comma-separated values (.csv) file that contains the data, choose **CSV**.

If you choose the **CSV** option, the report will include only a subset of the fields for each finding, approximately 45 fields that report key attributes of a finding. The fields include: *Finding Type, Title, Severity, Status, Description, First Seen, Last Seen, Fix Available, AWS account ID, Resource ID, Resource Tags, and Remediation*. These are in addition to fields that capture scoring details and reference URLs for each finding. The following is a sample of the CSV headers in a findings report:

```

AWSAccountId,Severity,Title,Findings,Code,ResourceId,ResourceType,ResourceName,Status,Description,FirstSeen,LastSeen,FixAvailable,AWSAccountID,ResourceID,ResourceTags,Remediation,UpdatedAt
Id,Type,Severity,Status,Description,FirstSeen,LastSeen,FixAvailable,AWSAccountID,ResourceID,ResourceTags,Remediation,UpdatedAt

```

- Under **Export location**, for **S3 URI**, specify the S3 bucket where you want to store the report:
 - To store the report in a bucket that your account owns, choose **Browse S3**. Amazon Inspector displays a table of the S3 buckets for your account. Select the row for the bucket that you want, and then choose **Choose**.

Tip

To also specify an Amazon S3 path prefix for the report, append a slash (/) and the prefix to the value in the **S3 URI** box. Amazon Inspector then includes the prefix when it adds the report to the bucket, and Amazon S3 generates the path specified by the prefix.

For example, if you want to use your AWS account ID as a prefix and your account ID is `111122223333`, append `/111122223333` to the value in the **S3 URI** box.

A *prefix* is similar to a directory path within an S3 bucket. It allows you to group similar objects together in a bucket, much like you might store similar files together in a folder on a file system. For more information, see [Organizing objects in the Amazon S3 console using folders](#) in the *Amazon Simple Storage Service User Guide*.

- To store the report in a bucket that another account owns, enter the URI for the bucket—for example, `s3://DOC-EXAMPLE_BUCKET`, where `DOC-EXAMPLE_BUCKET` is the name of the bucket. The bucket owner can find this information for you in the bucket's properties.
- For **KMS key**, specify the AWS KMS key that you want to use to encrypt the report:

- To use a key from your own account, choose the key from the list. The list displays customer managed, symmetric encryption KMS keys for your account.
- To use a key that another account owns, enter the Amazon Resource Name (ARN) of the key. The key owner can find this information for you in the key's properties. For more information, see [Finding the key ID and key ARN](#) in the *AWS Key Management Service Developer Guide*.

8. Choose **Export**.

Amazon Inspector generates the findings report, encrypts it with the KMS key that you specified, and adds it to the S3 bucket that you specified. Depending on the number of findings that you chose to include in the report, this process can take several minutes or hours. When the export is complete, Amazon Inspector displays a message indicating that your findings report was exported successfully. Optionally choose **View report** in the message to navigate to the report in Amazon S3.

Note that you can export only one report a time. If an export is currently in progress, wait until that export is complete before you try to export another report.

Troubleshoot export errors

If an error occurs when you try to export a findings report, Amazon Inspector displays a message describing the error. You can use the information in this topic as a guide to identify possible causes and solutions for the error.

For example, verify that the S3 bucket is in the current AWS Region and the bucket's policy allows Amazon Inspector to add objects to the bucket. Also verify that the AWS KMS key is enabled in the current Region, and ensure that the key policy allows Amazon Inspector to use the key.

After you address the error, try to export the report again.

Cannot have multiple reports error

If you are attempting to create a report but Amazon Inspector is already generating a report, you will receive an error stating **Reason: Cannot have multiple reports in-progress**. This error occurs because Amazon Inspector can only generate one report for an account at a time.

To resolve the error you can wait for the other report to finish or cancel it before requesting a new report.

You can check the status of a report by using the [GetFindingsReportStatus](#) operation, this operation returns the report ID of any report that is currently being generated.

If you need to, you can use the report ID given by the [GetFindingsReportStatus](#) operation to cancel a export that is currently in progress by using the [CancelFindingsReport](#) operation.

Creating custom responses to Amazon Inspector findings with Amazon EventBridge

Amazon Inspector creates an event in [Amazon EventBridge](#) for newly generated findings and aggregated findings. Amazon Inspector also creates an event for any changes to the state of a finding. This means Amazon Inspector creates a new event for a finding when you take actions like restarting a resource or changing tags associated with a resource. When Amazon Inspector creates a new event for an updated finding, the finding `id` stays the same.

Note

If your account is an Amazon Inspector delegated administrator account, EventBridge publishes events to your account and the member account where the events originated.

When using EventBridge events with Amazon Inspector, you can automate tasks to help you respond to security issues your findings reveal. To receive notifications about Amazon Inspector findings based on EventBridge events, you must create [an EventBridge rule](#) and specify a target for Amazon Inspector. The EventBridge rule allows EventBridge to send notifications for Amazon Inspector findings, and the target specifies where to send the notifications.

Amazon Inspector emits events to the default event bus in the AWS Region where you are currently using Amazon Inspector. This means you must configure event rules for each AWS Region where you activated Amazon Inspector and configured Amazon Inspector to receive EventBridge events. Amazon Inspector emits events on a best-effort basis.

This section provides you with an example of an event schema and describes how to create an EventBridge rule.

Event schema

The following is an example of the Amazon Inspector event format for an EC2 finding event. For example schema of other finding types and event types, see [EventBridge schema](#).

```

{
  "version": "0",
  "id": "66a7a279-5f92-971c-6d3e-c92da0950992",
  "detail-type": "Inspector2 Finding",
  "source": "aws.inspector2",
  "account": "111122223333",
  "time": "2023-01-19T22:46:15Z",
  "region": "us-east-1",
  "resources": ["i-0c2a343f1948d5205"],
  "detail": {
    "awsAccountId": "111122223333",
    "description": "\n It was discovered that the sound subsystem in the Linux
kernel contained a\n race condition in some situations. A local attacker could use
this to cause\n a denial of service (system crash).",
    "exploitAvailable": "YES",
    "exploitabilityDetails": {
      "lastKnownExploitAt": "Oct 24, 2022, 11:08:59 PM"
    },
    "findingArn": "arn:aws:inspector2:us-east-1:111122223333:finding/FINDING_ID",
    "firstObservedAt": "Jan 19, 2023, 10:46:15 PM",
    "fixAvailable": "YES",
    "lastObservedAt": "Jan 19, 2023, 10:46:15 PM",
    "packageVulnerabilityDetails": {
      "cvss": [{
        "baseScore": 4.7,
        "scoringVector": "CVSS:3.1/AV:L/AC:H/PR:L/UI:N/S:U/C:N/I:N/A:H",
        "source": "NVD",
        "version": "3.1"
      }],
      "referenceUrls": ["https://lore.kernel.org/all/
CAFc06XN7JDM4xSXGhtusQfS2mSBcx50VJKwQpCq=WeLt57aaZA@mail.gmail.com/", "https://
ubuntu.com/security/notices/USN-5792-1", "https://ubuntu.com/security/notices/
USN-5791-2", "https://ubuntu.com/security/notices/USN-5791-1", "https://ubuntu.com/
security/notices/USN-5793-2", "https://git.kernel.org/pub/scm/linux/kernel/git/
torvalds/linux.git/commit/?id=8423f0b6d513b259fdab9c9bf4aaa6188d054c2d", "https://
ubuntu.com/security/notices/USN-5793-1", "https://ubuntu.com/security/notices/
USN-5792-2", "https://ubuntu.com/security/notices/USN-5791-3", "https://ubuntu.com/
security/notices/USN-5793-4", "https://ubuntu.com/security/notices/USN-5793-3",
"https://git.kernel.org/linus/8423f0b6d513b259fdab9c9bf4aaa6188d054c2d(6.0-rc5)",
"https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2022-3303"],
      "relatedVulnerabilities": [],
      "source": "UBUNTU_CVE",

```

```

    "sourceUrl": "https://people.canonical.com/~ubuntu-security/cve/2022/
CVE-2022-3303.html",
    "vendorCreatedAt": "Sep 27, 2022, 11:15:00 PM",
    "vendorSeverity": "medium",
    "vulnerabilityId": "CVE-2022-3303",
    "vulnerablePackages": [{
      "arch": "X86_64",
      "epoch": 0,
      "fixedInVersion": "0:5.15.0.1027.31~20.04.16",
      "name": "linux-image-aws",
      "packageManager": "OS",
      "remediation": "apt update && apt install --only-upgrade linux-image-
aws",
      "version": "5.15.0.1026.30~20.04.16"
    }
  ],
  "remediation": {
    "recommendation": {
      "text": "None Provided"
    }
  },
  "resources": [{
    "details": {
      "awsEc2Instance": {
        "iamInstanceProfileArn": "arn:aws:iam::111122223333:instance-
profile/AmazonSSMRoleForInstancesQuickSetup",
        "imageId": "ami-0b7ff1a8d69f1bb35",
        "ipV4Addresses": ["172.31.85.212", "44.203.45.27"],
        "ipV6Addresses": [],
        "launchedAt": "Jan 19, 2023, 7:53:14 PM",
        "platform": "UBUNTU_20_04",
        "subnetId": "subnet-8213f2a3",
        "type": "t2.micro",
        "vpcId": "vpc-ab6650d1"
      }
    }
  },
  "id": "i-0c2a343f1948d5205",
  "partition": "aws",
  "region": "us-east-1",
  "type": "AWS_EC2_INSTANCE"
}],
"severity": "MEDIUM",
"status": "ACTIVE",
"title": "CVE-2022-3303 - linux-image-aws",

```

```
    "type": "PACKAGE_VULNERABILITY",
    "updatedAt": "Jan 19, 2023, 10:46:15 PM"
  }
}
```

Creating an EventBridge rule to notify you of Amazon Inspector findings

To increase the visibility of Amazon Inspector findings, you can use EventBridge to set up automated finding alerts that are sent to a messaging hub. This topic shows you how to send alerts for CRITICAL and HIGH severity findings to email, Slack, or Amazon Chime. You'll learn how to set up an Amazon Simple Notification Service topic and then connect that topic to an EventBridge event rule.

Step 1. Set up an Amazon SNS topic and endpoint

To set up automatic alerts, you must first set up a topic in Amazon Simple Notification Service and add an endpoint. For more information, refer to the [SNS guide](#).

This procedure establishes where you want to send Amazon Inspector findings data. The SNS topic can be added to an EventBridge event rule during or after the creation of the event rule.

Email setup

Creating an SNS topic

1. Sign in to the Amazon SNS console at <https://console.aws.amazon.com/sns/v3/home>.
2. From the navigation pane, select **Topics**, and then select **Create Topic**.
3. In the **Create topic** section, select **Standard**. Next, enter a topic name, such as **Inspector_to_Email**. Other details are optional.
4. Choose **Create Topic**. This opens a new panel with details for your new topic.
5. In the **Subscriptions** section, select **Create Subscription**.
6.
 - a. From the **Protocol** menu, select **Email**.
 - b. In the **Endpoint** field, enter the email address that you would like to receive notifications.

Note

You will be required to confirm your subscription through your email client after creating the subscription.

- c. Choose **Create subscription**.
7. Look for a subscription message in your inbox and choose **Confirm Subscription**.

Slack setup

Creating an SNS topic

1. Sign in to the Amazon SNS console at <https://console.aws.amazon.com/sns/v3/home>.
2. From the navigation pane, select **Topics**, and then select **Create Topic**.
3. In the **Create topic** section, select **Standard**. Next, enter a topic name, such as **Inspector_to_Slack**. Other details are optional. Choose **Create topic** to complete endpoint creation.

Configuring an AWS Chatbot client

1. Navigate to the AWS Chatbot console at <https://console.aws.amazon.com/chatbot/>.
2. From the **Configured clients** pane, select **Configure new client**.
3. Choose **Slack**, and then choose **Configure** to confirm.

Note

When choosing Slack, you must confirm permissions for AWS Chatbot to access your channel by selecting **allow**.

4. Select **Configure new channel** to open the configuration details pane.
 - a. Enter a name for the channel.
 - b. For **Slack channel**, choose the channel that you want to use.
 - c. In Slack, copy the channel ID of the private channel by right-clicking on the channel name and selecting **Copy Link**.

- d. On the AWS Management Console, in the AWS Chatbot window, paste the channel ID that you copied from Slack into the **Private channel ID** field.
 - e. In **Permissions**, choose to create an IAM role using a template if you do not already have a role.
 - f. For **Policy** templates, choose **Notification permissions**. This is the IAM policy template for AWS Chatbot. This policy provides the necessary read and list permissions for CloudWatch alarms, events, and logs, and for Amazon SNS topics.
 - g. For **Channel guardrail policies**, choose **AmazonInspector2ReadOnlyAccess**.
 - h. Choose the Region in which you previously created your SNS topic, and then select the Amazon SNS topic you created to send notifications to the Slack channel.
5. Select **Configure**.

Amazon Chime setup

Creating an SNS topic

1. Sign in to the Amazon SNS console at <https://console.aws.amazon.com/sns/v3/home>.
2. Select **Topics** from the navigation pane, and then select **Create Topic**.
3. In the **Create topic** section, select **Standard**. Next, enter a topic name, such as **Inspector_to_Chime**. Other details are optional. Choose **Create topic** to complete.

Configuring an AWS Chatbot client

1. Navigate to the AWS Chatbot console at <https://console.aws.amazon.com/chatbot/>.
2. From the **Configured clients** panel, select **Configure new client**.
3. Choose **Chime**, and then choose **Configure** to confirm.
4. From the **Configuration details** pane, enter a name for the channel.
5. In Amazon Chime, open the desired chat room.
 - a. Choose the gear icon in the upper-right corner and choose **Manage webhooks and bots**.
 - b. Select **Copy URL** to copy the webhook URL to your clipboard.
6. On the AWS Management Console, in the AWS Chatbot window, paste the URL you copied into the **Webhook URL** field.

7. In **Permissions**, choose to create an IAM role using a template if you do not already have a role.
8. For **Policy** templates, choose **Notification permissions**. This is the IAM policy template for AWS Chatbot. It provides the necessary read and list permissions for CloudWatch alarms, events, and logs, and for Amazon SNS topics.
9. Choose the Region in which you previously created your SNS topic, and then select the Amazon SNS topic you created to send notifications to the Amazon Chime room.
10. Select **Configure**.

Step 2. Create an EventBridge rule for Amazon Inspector findings

1. Sign in using your credentials.
2. Open the Amazon EventBridge console at <https://console.aws.amazon.com/events/>.
3. Select **Rules** from the navigation pane, and then select **Create rule**.
4. Enter a name and optional description for your rule.
5. Select **Rule with an event pattern** and then **Next**.
6. In the **Event Pattern** pane, choose **Custom patterns (JSON editor)**.
7. Paste the following JSON into the editor.

```
{
  "source": ["aws.inspector2"],
  "detail-type": ["Inspector2 Finding"],
  "detail": {
    "severity": ["HIGH", "CRITICAL"],
    "status": ["ACTIVE"]
  }
}
```

Note

This pattern sends notifications for any active CRITICAL or HIGH severity finding detected by Amazon Inspector.

Select **Next** when you are finished entering the event pattern.

8. On the **Select targets** page, choose **AWS service**. Then, for **Select target type**, choose **SNS topic**.
9. For **Topic**, select the name of the SNS topic you created in step 1. Then choose **Next**.
10. Add optional tags if needed and choose **Next**.
11. Review your rule and then choose **Create rule**.

EventBridge for Amazon Inspector multi-account environments

If you're an Amazon Inspector delegated administrator, EventBridge rules appear on your account based on applicable findings from your member accounts. If you set up findings notifications through EventBridge in your administrator account, as detailed in the preceding section, you'll receive notifications about multiple accounts. In other words, you'll be notified of findings and events generated by your member accounts in addition to those generated by your own account.

You can use the `accountId` from the finding's JSON details to identify the member account from which the Amazon Inspector finding originated.

Working with the dashboard in Amazon Inspector

The dashboard provides a snapshot of aggregated statistics for resources that Amazon Inspector scans. Use the dashboard to learn about coverage for your environment and critical findings.

Note

If your account is the delegated administrator account for an organization, the dashboard shows information for your account and every other account in the organization.

This section describes how to view the dashboard and understand the components that make up the dashboard.

Topics

- [Viewing the dashboard](#)
- [Understanding dashboard components and interpreting data](#)

Viewing the dashboard

The dashboard shows an overview of the coverage for your environment and critical findings.

To view the dashboard:

1. Sign in using your credentials, and then open the Amazon Inspector console at <https://console.aws.amazon.com/inspector/v2/home>.
2. From the navigation pane, choose **Dashboard**.
 - a. The dashboard refreshes data automatically every five minutes, you can refresh data manually by selecting the refresh icon at the top-right corner of the page.
 - b. You can view supporting data for an item by choosing the item.
 - c. If your account is the delegated administrator account for an organization, you can view aggregated statistics for a member account by entering the member account ID in the **Account** field.

Understanding dashboard components and interpreting data

Each section of the dashboard provides insight into key metrics and findings data, so you can understand the vulnerability posture of your AWS resources in your current AWS Region.

Environment coverage

The **Environment coverage** section provides statistics about the resources scanned by Amazon Inspector. In this section, you can see the count and percentage of Amazon EC2 instances, Amazon ECR images and AWS Lambda functions scanned by Amazon Inspector. If you manage multiple accounts through AWS Organizations as an Amazon Inspector delegated administrator, you will also see the total number of organization accounts, the number with Amazon Inspector activated, and the resulting coverage percentage for the organization. You can also use this section to determine which resources are not covered by Amazon Inspector. These resources may contain vulnerabilities that could be exploited to put your organization at risk. For more details, see [Assessing Amazon Inspector coverage of your AWS environment](#).

Choosing a coverage group takes you to the **Account management** page for the grouping you select. The account management page shows you details about which accounts, Amazon EC2 instances, and Amazon ECR repositories are covered by Amazon Inspector.

The following coverage groups are available:

- Account
- Instances
- Container repositories
- Container images
- Lambda

Critical findings

The **Critical findings** section provides a count of the critical vulnerabilities in your environment and a total count of all findings in your environment. In this section, the counts are shown per resource and assessment type. For more information about critical findings and how Amazon Inspector determines criticality, see [Understanding Amazon Inspector findings](#).

Choosing a critical finding group takes you to the **All findings** page and automatically applies filters to show all critical findings that match the grouping you selected.

The following critical finding groups are available:

- ECR container images findings
- Amazon EC2 findings
- Network reachability findings
- AWS Lambda function findings

Risk-based remediations

The **Risk-based remediations** section shows the top five software packages with critical vulnerabilities that affect the most resources in your environment. Remediating these packages can significantly reduce the number of critical risks to your environment. Choose the software package name to see associated vulnerability details and affected resources.

Accounts with the most critical findings

The **Accounts with the most critical findings** section shows the top five AWS accounts in your environment with the most critical findings, and the total number of findings for that account. This section is only viewable from the delegated administrator account when Amazon Inspector is configured for multi-account scanning with AWS Organizations. This view helps delegated administrators understand which accounts may be most at risk within the organization.

Choose **Account ID** to see more information about the affected member account.

Amazon ECR repositories with most critical findings

The **Elastic Container Registry (ECR) Repositories with most critical findings** section shows the top five Amazon ECR repositories in your environment with the most critical container image findings. The view shows the repository name, AWS account identifier, the repository creation date, number of critical vulnerabilities, and total number of vulnerabilities. This view helps you identify which repositories may be most at risk.

Choose **Repository name** to see more information about the affected repository.

Container images with most critical findings

The **Container images with most critical findings** section shows the top five container images in your environment with the most critical findings. The view shows image tag data, repository name, image digest, AWS account identifier, number of critical vulnerabilities, and total number of vulnerabilities. This view helps application owners identify which container images may need to be rebuilt and relaunched.

Choose **Container image** to see more information about the affected container image.

Instances with most critical findings

The **Instances with most critical findings** section shows the top five Amazon EC2 instances with the most critical findings. The view shows instance identifier, AWS account identifier, Amazon Machine Image (AMI) identifier, number of critical vulnerabilities, and total number of vulnerabilities. This view helps infrastructure owners identify which instances may require patching.

Choose **Instance ID** to see more information about the affected Amazon EC2 instance.

Amazon Machine Images (AMI) with most critical findings

The **Amazon Machine Images (AMIs) with most critical findings** section shows the top five AMIs in your environment with the most critical findings. The view shows the AMI identifier, AWS account identifier, number of affected EC2 instances running in the environment, the AMI creation date, the operating system platform of the AMI, the number of critical vulnerabilities, and the total number of vulnerabilities. This view helps infrastructure owners identify which AMIs may require rebuilding.

Choose **Affected instances** to see more information about the instances launched from the affected AMI.

AWS Lambda functions with most critical findings

The **AWS Lambda functions with most critical findings** section shows the top five Lambda functions in your environment with the most critical findings. The view shows the Lambda function name, AWS account identifier, runtime environment, the number of critical vulnerabilities, the number of high vulnerabilities, and the total number of vulnerabilities. This view helps infrastructure owners identify which Lambda functions may require remediation.

Choose **Function name** to see more information about the affected AWS Lambda function.

Searching the Amazon Inspector vulnerability database

You can search the Amazon Inspector vulnerability database for common vulnerabilities and exposures (CVE). Amazon Inspector uses information from the vulnerability database to produce details related to a CVE ID. You can view these details on the CVE details screen. Amazon Inspector tracks and produces [findings](#) for software vulnerabilities in the vulnerability database. Amazon Inspector only supports CVEs with platforms listed in the **Detection Platforms** section of the CVE details screen. This section describes how to search the Amazon Inspector vulnerability database using a CVE ID.

Note

Currently, CVE search doesn't support Microsoft Windows.

Searching the vulnerability database

This section describes how to search the vulnerability database in the console and with the Amazon Inspector API.

Note

You must activate Amazon Inspector in your current AWS Region before you can search the vulnerability database.

Console

1. Sign in using your credentials, and then open the Amazon Inspector console at <https://console.aws.amazon.com/inspector/v2/home>
2. From the navigation pane, choose **Vulnerability database search**.
3. In the search bar, enter a CVE ID, and choose **Search**.

API

Run the Amazon Inspector [SearchVulnerabilities](#) API, and provide a single CVE ID as `filterCriteria` in the following format: CVE-<year>-<ID>.

Understanding CVE details

This section describes how to interpret the CVE details page.

CVE details

The CVE details section includes the following information:

- CVE description and ID
- CVE Severity
- Common Vulnerability Scoring System (CVSS) and Exploit Prediction Scoring System (EPSS) scores
- Detection platforms

Note

If this field is empty, Amazon Inspector doesn't support detection for your CVE ID.

- Common Weakness Enumeration (CWE)
- Vendor created and updated dates

Vulnerability intelligence

The vulnerability intelligence section provides threat intelligence data like exploit targets and the last known public exploit date.

It also provides data from the Cybersecurity and Infrastructure Security Agency (CISA), which includes the remediation action, date the CVE was added to the Known Exploited Vulnerability catalog, and date time CISA expects federal agencies to remediate the CVE.

References

The references section provides links to resources for more information about the CVE.

Exporting SBOMs with Amazon Inspector

A software bill of materials (SBOM) is a nested inventory of all the open-source and third-party software components in your codebase. Amazon Inspector provides SBOMs for individual resources in your environment. You can use the Amazon Inspector console or Amazon Inspector API to generate SBOMs for your resources. You can export SBOMs for all resources that Amazon Inspector supports and monitors. Exported SBOMs provide information about your software supply. You can review the status of your resources by [assessing the coverage of your AWS environment](#). This section describes how to configure and export SBOMs.

Note

Currently, Amazon Inspector doesn't support exporting SBOMs for Windows Amazon EC2 instances.

Amazon Inspector formats

Amazon Inspector supports exporting SBOMs in **CycloneDX 1.4** and **SPDX 2.3** compatible formats. Amazon Inspector exports SBOMs as JSON files to the Amazon S3 bucket you choose.

Note

SPDX format exports from Amazon Inspector are compatible with systems using **SPDX 2.3**, however they don't contain the Creative Commons Zero (CC0) field. This is because including this field would allow users to redistribute or edit the material.

Example of CycloneDX 1.4 SBOM format from Amazon Inspector

```
{
  "bomFormat": "CycloneDX",
  "specVersion": "1.4",
  "version": 1,
  "metadata": {
    "timestamp": "2023-06-02T01:17:46Z",
    "component": null,
  }
}
```

```

    "properties": [
      {
        "name": "imageId",
        "value":
"sha256:c8ee97f7052776ef223080741f61fcdf6a3a9107810ea9649f904aa4269fdac6"
      },
      {
        "name": "architecture",
        "value": "arm64"
      },
      {
        "name": "accountId",
        "value": "111122223333"
      },
      {
        "name": "resourceType",
        "value": "AWS_ECR_CONTAINER_IMAGE"
      }
    ]
  },
  "components": [
    {
      "type": "library",
      "name": "pip",
      "purl": "pkg:pypi/pip@22.0.4?path=usr/local/lib/python3.8/site-packages/
pip-22.0.4.dist-info/METADATA",
      "bom-ref": "98dc550d1e9a0b24161daaa0d535c699"
    },
    {
      "type": "application",
      "name": "libss2",
      "purl": "pkg:dpkg/libss2@1.44.5-1+deb10u3?
arch=ARM64&epoch=0&upstream=libss2-1.44.5-1+deb10u3.src.dpkg",
      "bom-ref": "2f4d199d4ef9e2ae639b4f8d04a813a2"
    },
    {
      "type": "application",
      "name": "liblz4-1",
      "purl": "pkg:dpkg/liblz4-1@1.8.3-1+deb10u1?
arch=ARM64&epoch=0&upstream=liblz4-1-1.8.3-1+deb10u1.src.dpkg",
      "bom-ref": "9a6be8907ead891b070e60f5a7b7aa9a"
    },
    {
      "type": "application",

```



```

    "name": "mawk",
    "purl": "pkg:dpkg/mawk@1.3.3-17+b3?
arch=ARM64&epoch=0&upstream=mawk-1.3.3-17+b3.src.dpkg",
    "bom-ref": "c2015852a729f97fde924e62a16f78a5"
  },
  {
    "type": "application",
    "name": "libgmp10",
    "purl": "pkg:dpkg/libgmp10@6.1.2+dfsg-4+deb10u1?
arch=ARM64&epoch=2&upstream=libgmp10-6.1.2+dfsg-4+deb10u1.src.dpkg",
    "bom-ref": "52907290f5beef00dff8da77901b1085"
  },
  {
    "type": "application",
    "name": "ncurses-bin",
    "purl": "pkg:dpkg/ncurses-bin@6.1+20181013-2+deb10u3?
arch=ARM64&epoch=0&upstream=ncurses-bin-6.1+20181013-2+deb10u3.src.dpkg",
    "bom-ref": "cd20cfb9ebeeada3809764376f43bce"
  }
],
"vulnerabilities": [
  {
    "id": "CVE-2022-40897",
    "affects": [
      {
        "ref": "a74a4862cc654a2520ec56da0c81cdb3"
      },
      {
        "ref": "0119eb286405d780dc437e7dbf2f9d9d"
      }
    ]
  }
]
}

```

Example of SPDX 2.3 SBOM format from Amazon Inspector

```

{
  "name": "409870544328/EC2/i-022fba820db137c64/ami-074ea14c08effb2d8",
  "spdxVersion": "SPDX-2.3",
  "creationInfo": {

```

```

"created": "2023-06-02T21:19:22Z",
"creators": [
  "Organization: 409870544328",
  "Tool: Amazon Inspector SBOM Generator"
],
"documentNamespace": "EC2://i-022fba820db137c64/AMAZON_LINUX_2/null/x86_64",
"comment": "",
"packages": [{
  "name": "elfutils-libelf",
  "versionInfo": "0.176-2.amzn2",
  "downloadLocation": "NOASSERTION",
  "sourceInfo": "/var/lib/rpm/Packages",
  "filesAnalyzed": false,
  "externalRefs": [{
    "referenceCategory": "PACKAGE-MANAGER",
    "referenceType": "purl",
    "referenceLocator": "pkg:rpm/elfutils-libelf@0.176-2.amzn2?
arch=X86_64&epoch=0&upstream=elfutils-libelf-0.176-2.amzn2.src.rpm"
  }],
  "SPDXID": "SPDXRef-Package-rpm-elfutils-libelf-ddf56a513c0e76ab2ae3246d9a91c463"
},
{
  "name": "libcurl",
  "versionInfo": "7.79.1-1.amzn2.0.1",
  "downloadLocation": "NOASSERTION",
  "sourceInfo": "/var/lib/rpm/Packages",
  "filesAnalyzed": false,
  "externalRefs": [{
    "referenceCategory": "PACKAGE-MANAGER",
    "referenceType": "purl",
    "referenceLocator": "pkg:rpm/libcurl@7.79.1-1.amzn2.0.1?
arch=X86_64&epoch=0&upstream=libcurl-7.79.1-1.amzn2.0.1.src.rpm"
  }],
  "referenceCategory": "SECURITY",
  "referenceType": "vulnerability",
  "referenceLocator": "CVE-2022-32205"
}],
"SPDXID": "SPDXRef-Package-rpm-libcurl-710fb33829bc5106559bcd380cddb7d5"
},
{
  "name": "hunspell-en-US",

```

```

"versionInfo": "0.20121024-6.amzn2.0.1",
"downloadLocation": "NOASSERTION",
"sourceInfo": "/var/lib/rpm/Packages",
"filesAnalyzed": false,
"externalRefs": [{
  "referenceCategory": "PACKAGE-MANAGER",
  "referenceType": "purl",
  "referenceLocator": "pkg:rpm/hunspell-en-US@0.20121024-6.amzn2.0.1?
arch=NOARCH&epoch=0&upstream=hunspell-en-US-0.20121024-6.amzn2.0.1.src.rpm"
}],
"SPDXID": "SPDXRef-Package-rpm-hunspell-en-US-de19ae0883973d6cea5e7e079d544fe5"
},
{
  "name": "grub2-tools-minimal",
  "versionInfo": "2.06-2.amzn2.0.6",
  "downloadLocation": "NOASSERTION",
  "sourceInfo": "/var/lib/rpm/Packages",
  "filesAnalyzed": false,
  "externalRefs": [{
    "referenceCategory": "PACKAGE-MANAGER",
    "referenceType": "purl",
    "referenceLocator": "pkg:rpm/grub2-tools-minimal@2.06-2.amzn2.0.6?
arch=X86_64&epoch=1&upstream=grub2-tools-minimal-2.06-2.amzn2.0.6.src.rpm"
  }],
  "SPDXID": "SPDXRef-Package-rpm-grub2-tools-minimal-c56b7ea76e5a28ab8f232ef6d7564636"
},
{
  "name": "unixODBC-devel",
  "versionInfo": "2.3.1-14.amzn2",
  "downloadLocation": "NOASSERTION",
  "sourceInfo": "/var/lib/rpm/Packages",
  "filesAnalyzed": false,
  "externalRefs": [{
    "referenceCategory": "PACKAGE-MANAGER",
    "referenceType": "purl",
    "referenceLocator": "pkg:rpm/unixODBC-devel@2.3.1-14.amzn2?
arch=X86_64&epoch=0&upstream=unixODBC-devel-2.3.1-14.amzn2.src.rpm"
  }],

```

```

    "SPDXID": "SPDXRef-Package-rpm-unixODBC-devel-1bb35add92978df021a13fc9f81237d2"
  }
],
"relationships": [{
  "spdxElementId": "SPDXRef-DOCUMENT",
  "relatedSpdxElement": "SPDXRef-Package-rpm-elfutils-libelf-
ddf56a513c0e76ab2ae3246d9a91c463",
  "relationshipType": "DESCRIBES"
},
{
  "spdxElementId": "SPDXRef-DOCUMENT",
  "relatedSpdxElement": "SPDXRef-Package-rpm-yajl-8476ce2db98b28cfab2b4484f84f1903",
  "relationshipType": "DESCRIBES"
},
{
  "spdxElementId": "SPDXRef-DOCUMENT",
  "relatedSpdxElement": "SPDXRef-Package-rpm-unixODBC-
devel-1bb35add92978df021a13fc9f81237d2",
  "relationshipType": "DESCRIBES"
}
],
"SPDXID": "SPDXRef-DOCUMENT"
}

```

Filters for SBOMs

When you export SBOMs you can include filters to create reports for specific subsets of resources. If you don't supply a filter the SBOMs for all active, supported resources are exported. And if you are a delegated administrator this includes resources for all members too. The following filters are available:

- **AccountID** — This filter can be used to export SBOMs for any resources associated with specific Account ID.
- **EC2 instance tag** — This filter can be used to export SBOMs for EC2 instances with specific tags.
- **Function name** — This filter can be used to export SBOMs for specific Lambda functions.
- **Image tag** — This filter can be used to export SBOMs for container images with specific tags.
- **Lambda function tag** — This filter can be used to export SBOMs for Lambda functions with specific tags.
- **Resource type** — This filter can be used to filter resource type: EC2/ECR/Lambda.

- **Resource ID** — This filter can be used to export an SBOM for a specific resource.
- **Repository name** — This filter can be used to generate SBOMs for container images in specific repositories.

Configure and export SBOMs

To export SBOMs, you must first configure an Amazon S3 bucket and a AWS KMS key that Amazon Inspector is allowed to use. You can use filters to export SBOMs for specific subsets of your resources. To export SBOMs for multiple accounts in an AWS Organization, follow these steps while signed in as the Amazon Inspector delegated administrator.

Prerequisites

- Supported resources that are being actively monitored by Amazon Inspector.
- An Amazon S3 bucket configured with a policy that allows Amazon Inspector to add object to. For information on configuring the policy see [Configure export permissions](#).
- An AWS KMS key configured with a policy that allows Amazon Inspector to use to encrypt your reports. For information on configuring the policy see [Configure an AWS KMS key for export](#).

Note

If you have previously configured an Amazon S3 bucket and an AWS KMS key for [findings export](#) you can use the same bucket and key for SBOM export.

Choose your preferred access method to export an SBOM.

Console

1. Sign in using your credentials, and then open the Amazon Inspector console at <https://console.aws.amazon.com/inspector/v2/home>.
2. Using the AWS Region selector in the upper-right corner of the page, select the Region with the resources you want to export SBOM for.
3. In the navigation pane, choose **Export SBOMs**.
4. (Optional) In the **Export SBOMs** page, use the **Add filter** menu to select a subset of resources to create reports for. If no filter is provided Amazon Inspector will export reports

for all active resources. If you are a delegated administrator this will include all active resources in your organization.

5. Under **Export setting** select the format you want for the SBOM.
6. Enter an **Amazon S3 URI** or choose **Browse Amazon S3** to select an Amazon S3 location to store the SBOM.
7. Enter a **AWS KMS key** configured for Amazon Inspector to use to encrypt your reports.

API

- To export SBOMs for your resources programmatically, use the [CreateSbomExport](#) operation of the Amazon Inspector API.

In your request, use the `reportFormat` parameter to specify the SBOM output format, choose `CYCLONEDX_1_4` or `SPDX_2_3`. The `s3Destination` parameter is required and you must specify an S3 bucket configured with a policy that allows Amazon Inspector to write to it. Optionally use `resourceFilterCriteria` parameters to limit the scope of the report to specific resources.

AWS CLI

- To export SBOMs for your resources using the AWS Command Line Interface run the following command:

```
aws inspector2 create-sbom-export --report-format  
FORMAT --s3-destination bucketName=amzn-s3-demo-  
bucket1,keyPrefix=PREFIX,kmsKeyArn=arn:aws:kms:Region:111122223333:key/123
```

In your request, replace *FORMAT* with the format of your choice, `CYCLONEDX_1_4` or `SPDX_2_3`. Then replace the *user input placeholders* for the s3 destination with the name of the S3 bucket to export to, the prefix to use for the output in S3, and the ARN for the KMS key you are using to encrypt the reports.

Amazon EventBridge event schema for Amazon Inspector events

[Amazon EventBridge](#) delivers a stream of real-time data from applications and other AWS services to targets, such as AWS Lambda functions, Amazon Simple Notification Service topics, and data streams in Amazon Kinesis Data Streams. To support integration with other applications, services, and systems, Amazon Inspector automatically publishes findings to EventBridge as [events](#). You can use Amazon Inspector to publish events for findings, coverage, and scans. This section provides example schemas for EventBridge events.

Topics

- [Amazon EventBridge base schema for Amazon Inspector](#)
- [Amazon Inspector finding event schema example](#)
- [Amazon Inspector initial scan complete event schema example](#)
- [Amazon Inspector coverage event schema example](#)

Amazon EventBridge base schema for Amazon Inspector

The following is an example of the basic schema for an EventBridge event for Amazon Inspector. Event details differ based on the type of event.

```
{
  "version": "0",
  "id": "Event ID",
  "detail-type": "Inspector2 *event type*",
  "source": "aws.inspector2",
  "account": "AWS account ID (string)",
  "time": "event timestamp (string)",
  "region": "AWS Region (string)",
  "resources": [
    *IDs or ARNs of the resources involved in the event*
  ],
  "detail": {
    *Details of an Amazon Inspector event type*
  }
}
```

Amazon Inspector finding event schema example

The following includes examples of the schema for an EventBridge event for Amazon Inspector findings. Finding events are created when Amazon Inspector identifies a software vulnerability or network issue in one of your resources. For a guide to creating notifications in response to this type of event, see [Creating custom responses to Amazon Inspector findings with Amazon EventBridge](#).

The following fields identify a finding event:

- `detail-type` is set to `Inspector2 Finding`.
- `detail` describes the finding.
- `detail.resources.tags` is where key-value data is stored.

You can filter the tabs to see finding event schemas for different resources and finding types.

Amazon EC2 package vulnerability finding

```
{
  "version": "0",
  "id": "4d621919-f1f4-4201-a0e2-37e4e330ff51",
  "detail-type": "Inspector2 Finding",
  "source": "aws.inspector2",
  "account": "123456789012",
  "time": "2024-09-04T17:00:36Z",
  "region": "eu-central-1",
  "resources": [
    "i-12345678901234567"
  ],
  "detail": {
    "awsAccountId": "123456789012",
    "description": "In snapd versions prior to 2.62, snapd failed to properly check the destination of symbolic links when extracting a snap. The snap format is a squashfs file-system image and so can contain symbolic links and other file types. Various file entries within the snap squashfs image (such as icons and desktop files etc) are directly read by snapd when it is extracted. An attacker who could convince a user to install a malicious snap which contained symbolic links at these paths could then cause snapd to write out the contents of the symbolic link destination into a world-readable directory. This in-turn could allow an unprivileged user to gain access to privileged information.",
    "epss": {
```



```
    "score": 0.00043
  },
  "exploitAvailable": "NO",
  "findingArn": "arn:aws:inspector2:eu-
central-1:123456789012:finding/FINDING_ID",
  "firstObservedAt": "Wed Sep 04 16:59:44.356 UTC 2024",
  "fixAvailable": "YES",
  "inspectorScore": 4.8,
  "inspectorScoreDetails": {
    "adjustedCvss": {
      "adjustments": [],
      "cvssSource": "UBUNTU_CVE",
      "score": 4.8,
      "scoreSource": "UBUNTU_CVE",
      "scoringVector": "CVSS:3.1/AV:L/AC:L/PR:L/UI:R/S:U/C:L/I:L/A:L",
      "version": "3.1"
    }
  },
  "lastObservedAt": "Wed Sep 04 16:59:44.476 UTC 2024",
  "packageVulnerabilityDetails": {
    "cvss": [
      {
        "baseScore": 4.8,
        "scoringVector": "CVSS:3.1/AV:L/AC:L/PR:L/UI:R/S:U/C:L/I:L/A:L",
        "source": "UBUNTU_CVE",
        "version": "3.1"
      },
      {
        "baseScore": 7.3,
        "scoringVector": "CVSS:3.1/AV:L/AC:L/PR:L/UI:R/S:U/C:H/I:H/A:H",
        "source": "NVD",
        "version": "3.1"
      }
    ],
    "referenceUrls": [
      "https://www.cve.org/CVERecord?id=CVE-2024-29069",
      "https://ubuntu.com/security/notices/USN-6940-1"
    ],
    "relatedVulnerabilities": [
      "USN-6940-1"
    ],
    "source": "UBUNTU_CVE",
    "sourceUrl": "https://people.canonical.com/~ubuntu-security/cve/2024/
CVE-2024-29069.html",
```

```
"vendorCreatedAt": "Thu Jul 25 20:15:00.000 UTC 2024",
"vendorSeverity": "medium",
"vulnerabilityId": "CVE-2024-29069",
"vulnerablePackages": [
  {
    "arch": "ALL",
    "epoch": 0,
    "fixedInVersion": "0:2.63+22.04ubuntu0.1",
    "name": "snapd",
    "packageManager": "OS",
    "remediation": "apt-get update && apt-get upgrade",
    "version": "2.63"
  }
],
"remediation": {
  "recommendation": {
    "text": "None Provided"
  }
},
"resources": [
  {
    "details": {
      "awsEc2Instance": {
        "iamInstanceProfileArn":
"arn:aws:iam::123456789012:instance-profile/AmazonSSMRoleForInstancesQuickSetup",
        "imageId": "ami-02ff980600c693b38",
        "ipV4Addresses": [
          "1.23.456.789",
          "123.45.67.890"
        ],
        "ipV6Addresses": [],
        "launchedAt": "Wed Sep 04 16:57:40.000 UTC 2024",
        "platform": "UBUNTU_22_04",
        "subnetId": "subnet-12345678",
        "type": "t2.small",
        "vpcId": "vpc-12345678"
      }
    },
    "id": "i-12345678901234567",
    "partition": "aws",
    "region": "eu-central-1",
    "type": "AWS_EC2_INSTANCE"
  }
]
```

```

    ],
    "severity": "MEDIUM",
    "status": "CLOSED",
    "title": "CVE-2024-29069 - snapd",
    "type": "PACKAGE_VULNERABILITY",
    "updatedAt": "Wed Sep 04 17:00:36.951 UTC 2024"
  }
}

```

Amazon EC2 network reachability finding

```

{
  "version": "0",
  "id": "9eb1603b-4263-19ec-8be2-33184694cb92",
  "detail-type": "Inspector2 Finding",
  "source": "aws.inspector2",
  "account": "123456789012",
  "time": "2024-09-05T13:06:56Z",
  "region": "eu-central-1",
  "resources": ["i-12345678901234567"],
  "detail": {
    "awsAccountId": "123456789012",
    "description": "On the instance i-12345678901234567, the port range 22-22 is reachable from the InternetGateway igw-261bab4d from an attached ENI eni-094ad651219472857.",
    "findingArn": "arn:aws:inspector2:eu-central-1:123456789012:finding/FINDING_ID",
    "firstObservedAt": "Thu Sep 05 13:06:56.334 UTC 2024",
    "lastObservedAt": "Thu Sep 05 13:06:56.334 UTC 2024",
    "networkReachabilityDetails": {
      "networkPath": {
        "steps": [{
          "componentId": "igw-261bab4d",
          "componentType": "AWS::EC2::InternetGateway"
        }, {
          "componentId": "acl-171b527d",
          "componentType": "AWS::EC2::NetworkAcl"
        }, {
          "componentId": "sg-0d34debf87410f2d9",
          "componentType": "AWS::EC2::SecurityGroup"
        }, {

```

```

        "componentId": "eni-094ad651219472857",
        "componentType": "AWS::EC2::NetworkInterface"
    }, {
        "componentId": "i-12345678901234567",
        "componentType": "AWS::EC2::Instance"
    }
  ]
},
"openPortRange": {
  "begin": 22,
  "end": 22
},
"protocol": "TCP"
},
"remediation": {
  "recommendation": {
    "text": "You can restrict access to your instance by modifying the
Security Groups or ACLs in the network path."
  }
},
"resources": [{
  "details": {
    "awsEc2Instance": {
      "iamInstanceProfileArn": "arn:aws:iam::123456789012:instance-
profile/AmazonSSMRoleForInstancesQuickSetup",
      "imageId": "ami-02ff980600c693b38",
      "ipV4Addresses": ["1.23.456.789", "123.45.67.890"],
      "ipV6Addresses": [],
      "launchedAt": "Wed Sep 04 17:41:24.000 UTC 2024",
      "platform": "UBUNTU_22_04",
      "subnetId": "subnet-12345678",
      "type": "t2.small",
      "vpcId": "vpc-12345678"
    }
  },
  "id": "i-12345678901234567",
  "partition": "aws",
  "region": "eu-central-1",
  "type": "AWS_EC2_INSTANCE"
}],
"severity": "MEDIUM",
"status": "ACTIVE",
"title": "Port 22 is reachable from an Internet Gateway - TCP",
"type": "NETWORK_REACHABILITY",
"updatedAt": "Thu Sep 05 13:06:56.334 UTC 2024"

```

```
}
}
```

Amazon ECR package vulnerability finding

```
{
  "version": "0",
  "id": "5325facf-a1aa-7d97-6bce-25fde6f6d2fc",
  "detail-type": "Inspector2 Finding",
  "source": "aws.inspector2",
  "account": "123456789012",
  "time": "2024-09-04T16:55:38Z",
  "region": "eu-central-1",
  "resources": [
    "arn:aws:ecr:eu-central-1:123456789012:repository/inspector2/
    sha256:84f507df33c6864d49c296fb734192696e4cb6f78166ac51ac8b9b118181085d"
  ],
  "detail.resources.tags.testkey": "allow",
  "detail": {
    "awsAccountId": "123456789012",
    "description": "Possible denial of service in X.509 name checks",
    "epss": {
      "score": 0.00045
    },
    "exploitAvailable": "NO",
    "findingArn": "arn:aws:inspector2:eu-
    central-1:123456789012:finding/FINDING_ID",
    "firstObservedAt": "Wed Sep 04 16:55:38.411 UTC 2024",
    "fixAvailable": "YES",
    "lastObservedAt": "Wed Sep 04 16:55:38.411 UTC 2024",
    "packageVulnerabilityDetails": {
      "cvss": [],
      "referenceUrls": [
        "https://www.cve.org/CVERecord?id=CVE-2024-6119",
        "https://ubuntu.com/security/notices/USN-6986-1"
      ],
      "relatedVulnerabilities": [
        "USN-6986-1"
      ],
      "source": "UBUNTU_CVE",

```

```

    "sourceUrl": "https://people.canonical.com/~ubuntu-security/cve/2024/
CVE-2024-6119.html",
    "vendorCreatedAt": "Tue Sep 03 00:00:00.000 UTC 2024",
    "vendorSeverity": "medium",
    "vulnerabilityId": "CVE-2024-6119",
    "vulnerablePackages": [
      {
        "arch": "ARM64",
        "epoch": 0,
        "fixedInVersion": "0:3.0.13-0ubuntu3.4",
        "name": "libssl3t64",
        "packageManager": "OS",
        "release": "0ubuntu3.2",
        "remediation": "apt-get update && apt-get upgrade",
        "sourceLayerHash":
"sha256:1567e7ea90b67fc95ccdeeeec39bdc3045098dee7e0c604975b957a9f8c0e9616",
        "version": "3.0.13"
      },
      {
        "arch": "ARM64",
        "epoch": 0,
        "fixedInVersion": "0:3.0.13-0ubuntu3.4",
        "name": "openssl",
        "packageManager": "OS",
        "release": "0ubuntu3.2",
        "remediation": "apt-get update && apt-get upgrade",
        "sourceLayerHash":
"sha256:1567e7ea90b67fc95ccdeeeec39bdc3045098dee7e0c604975b957a9f8c0e9616",
        "version": "3.0.13"
      }
    ]
  },
  "remediation": {
    "recommendation": {
      "text": "None Provided"
    }
  },
  "resources": [
    {
      "details": {
        "awsEcrContainerImage": {
          "architecture": "arm64",
          "imageHash":
"sha256:84f507df33c6864d49c296fb734192696e4cb6f78166ac51ac8b9b118181085d",

```

```

        "imageTags": [
            "ubuntu_latest"
        ],
        "platform": "UBUNTU_24_04",
        "pushedAt": "Wed Sep 04 16:55:28.000 UTC 2024",
        "registry": "123456789012",
        "repositoryName": "inspector2"
    }
},
    "id": "arn:aws:ecr:eu-central-1:123456789012:repository/inspector2/
sha256:84f507df33c6864d49c296fb734192696e4cb6f78166ac51ac8b9b118181085d",
    "partition": "aws",
    "region": "eu-central-1",
    "type": "AWS_ECR_CONTAINER_IMAGE"
}
],
"severity": "MEDIUM",
"status": "ACTIVE",
"title": "CVE-2024-6119 - libssl3t64, openssl",
"type": "PACKAGE_VULNERABILITY",
"updatedAt": "Wed Sep 04 16:55:38.411 UTC 2024"
}
}

```

Lambda package vulnerability finding

```

{
    "version": "0",
    "id": "9eadd71a-e49c-9864-6ba9-2a5d3f83c88f",
    "detail-type": "Inspector2 Finding",
    "source": "aws.inspector2",
    "account": "123456789012",
    "time": "2024-09-04T16:50:37Z",
    "region": "eu-central-1",
    "resources": [
        "arn:aws:lambda:eu-central-1:123456789012:function:VulnerableFunction:
$LATEST"
    ],
    "detail": {
        "awsAccountId": "123456789012",

```

```

    "description": "Flask is a lightweight WSGI web application framework. When
all of the following conditions are met, a response containing data intended for
one client may be cached and subsequently sent by the proxy to other clients. If
the proxy also caches `Set-Cookie` headers, it may send one client's `session`
cookie to other clients. The severity depends on the application's use of the
session and the proxy's behavior regarding cookies. The risk depends on all these
conditions being met.\n\n1. The application must be hosted behind a caching proxy
that does not strip cookies or ignore responses with cookies. 2. The application
sets `session.permanent = True` 3. The application does not access or modify the
session at any point during a request. 4. `SESSION_REFRESH_EACH_REQUEST` enabled
(the default). 5. The application does not set a `Cache-Control` header to indicate
that a page is private or should not be cached.\n\nThis happens because vulnerable
versions of Flask only set the `Vary: Cookie` header when the session is ac",
    "epss": {
        "score": 0.00208
    },
    "exploitAvailable": "YES",
    "exploitabilityDetails": {
        "lastKnownExploitAt": "Sat Aug 31 00:04:50.000 UTC 2024"
    },
    "findingArn": "arn:aws:inspector2:eu-
central-1:123456789012:finding/FINDING_ID",
    "firstObservedAt": "Wed Sep 04 16:50:37.627 UTC 2024",
    "fixAvailable": "YES",
    "inspectorScore": 7.5,
    "inspectorScoreDetails": {
        "adjustedCvss": {
            "cvssSource": "NVD",
            "score": 7.5,
            "scoreSource": "NVD",
            "scoringVector": "CVSS:3.1/AV:N/AC:L/PR:N/UI:N/S:U/C:H/I:N/A:N",
            "version": "3.1"
        }
    },
    "lastObservedAt": "Wed Sep 04 16:50:37.627 UTC 2024",
    "packageVulnerabilityDetails": {
        "cvss": [
            {
                "baseScore": 7.5,
                "scoringVector": "CVSS:3.1/AV:N/AC:L/PR:N/UI:N/S:U/C:H/I:N/A:N",
                "source": "NVD",
                "version": "3.1"
            }
        ]
    },

```



```

    "referenceUrls": [
      "https://www.debian.org/security/2023/dsa-5442",
      "https://lists.debian.org/debian-lts-announce/2023/08/msg00024.html"
    ],
    "relatedVulnerabilities": [],
    "source": "NVD",
    "sourceUrl": "https://nvd.nist.gov/vuln/detail/CVE-2023-30861",
    "vendorCreatedAt": "Tue May 02 18:15:52.000 UTC 2023",
    "vendorSeverity": "HIGH",
    "vendorUpdatedAt": "Sun Aug 20 21:15:09.000 UTC 2023",
    "vulnerabilityId": "CVE-2023-30861",
    "vulnerablePackages": [
      {
        "epoch": 0,
        "filePath": "requirements.txt",
        "fixedInVersion": "2.3.2",
        "name": "flask",
        "packageManager": "PIP",
        "version": "2.0.0"
      }
    ]
  },
  "remediation": {
    "recommendation": {
      "text": "None Provided"
    }
  },
  "resources": [
    {
      "details": {
        "awsLambdaFunction": {
          "architectures": [
            "X86_64"
          ],
          "codeSha256": "07jkFEmfPB+CK3Y6Pby5zW9gjG
+zusAaqRRMGS8B27c=",
          "executionRoleArn": "arn:aws:iam::123456789012:role/service-
role/VulnerableFunction-role-f9vs5mq8",
          "functionName": "VulnerableFunction",
          "lastModifiedAt": "Wed Sep 04 16:50:20.000 UTC 2024",
          "packageType": "ZIP",
          "runtime": "PYTHON_3_11",
          "version": "$LATEST"
        }
      }
    }
  ]
}

```

```

        },
        "id": "arn:aws:lambda:eu-
central-1:123456789012:function:VulnerableFunction:$LATEST",
        "partition": "aws",
        "region": "eu-central-1",
        "type": "AWS_LAMBDA_FUNCTION"
    }
],
"severity": "HIGH",
"status": "ACTIVE",
"title": "CVE-2023-30861 - flask",
"type": "PACKAGE_VULNERABILITY",
"updatedAt": "Wed Sep 04 16:50:37.627 UTC 2024"
}
}

```

Lambda code vulnerability finding

```

{
  "version": "0",
  "id": "e764f7be-f931-ff1b-204b-8cab2d91724b",
  "detail-type": "Inspector2 Finding",
  "source": "aws.inspector2",
  "account": "123456789012",
  "time": "2024-09-04T16:51:01Z",
  "region": "eu-central-1",
  "resources": [
    "arn:aws:lambda:eu-central-1:123456789012:function:VulnerableFunction:
$LATEST"
  ],
  "detail": {
    "awsAccountId": "123456789012",
    "codeVulnerabilityDetails": {
      "cwes": [
        "CWE-798"
      ],
      "detectorId": "python/hardcoded-credentials@v1.0",
      "detectorName": "Hardcoded credentials",
      "detectorTags": [
        "secrets",
        "security",

```

```

        "owasp-top10",
        "top25-cwes",
        "cwe-798",
        "Python"
    ],
    "filePath": {
        "endLine": 6,
        "fileName": "lambda_function.py",
        "filePath": "lambda_function.py",
        "startLine": 6
    },
    "ruleId": "python-detect-hardcoded-aws-credentials"
},
"description": "Access credentials, such as passwords and access keys,
should not be hardcoded in source code. Hardcoding credentials may cause leaks even
after removing them. This is because version control systems might retain older
versions of the code. Credentials should be stored securely and obtained from the
runtime environment.",
"findingArn": "arn:aws:inspector2:eu-
central-1:123456789012:finding/FINDING_ID",
"firstObservedAt": "Wed Sep 04 16:51:01.869 UTC 2024",
"lastObservedAt": "Wed Sep 04 16:51:01.869 UTC 2024",
"remediation": {
    "recommendation": {
        "text": "Your code uses hardcoded AWS credentials which might
allow unauthorized users access to your AWS account. These attacks can occur
a long time after the credentials are removed from the code. We recommend that
you set AWS credentials with environment variables or an AWS profile instead.
You should consider deleting the affected account or rotating the secret key
and then monitoring Amazon CloudWatch for unexpected activity.\n[https://
boto3.amazonaws.com/v1/documentation/api/latest/guide/credentials.html](https://
boto3.amazonaws.com/v1/documentation/api/latest/guide/credentials.html)"
    }
},
"resources": [
    {
        "details": {
            "awsLambdaFunction": {
                "architectures": [
                    "X86_64"
                ],
                "codeSha256": "07jkFEmfPB+CK3Y6Pby5zW9gjG
+zusAaqRRMGS8B27c=",

```

```

        "executionRoleArn": "arn:aws:iam::123456789012:role/service-
role/VulnerableFunction-role-f9vs5mq8",
        "functionName": "VulnerableFunction",
        "lastModifiedAt": "Wed Sep 04 16:50:20.000 UTC 2024",
        "packageType": "ZIP",
        "runtime": "PYTHON_3_11",
        "version": "$LATEST"
    }
},
    "id": "arn:aws:lambda:eu-
central-1:123456789012:function:VulnerableFunction:$LATEST",
    "partition": "aws",
    "region": "eu-central-1",
    "type": "AWS_LAMBDA_FUNCTION"
}
],
"severity": "CRITICAL",
"status": "ACTIVE",
"title": "CWE-798 - Hardcoded credentials",
"type": "CODE_VULNERABILITY",
"updatedAt": "Wed Sep 04 16:51:01.869 UTC 2024"
}
}

```

Note

The detail value returns the JSON details of a single finding as an object. It does not return the entire findings response syntax, which supports multiple findings within an array.

Amazon Inspector initial scan complete event schema example

The following is an example of the EventBridge event schema for an Amazon Inspector event for completing an initial scan. This event is created when Amazon Inspector completes an initial scan of one of your resources.

The following fields identify an initial scan complete event:

- The detail-type field is set to Inspector2 Scan.

- The detail object contains a finding-severity-counts object that details the number of findings in the applicable severity categories, such as CRITICAL, HIGH, and MEDIUM.

Select from the options to see different initial scan event schemas by resource type.

Amazon EC2 instance initial scan

```
{
  "version": "0",
  "id": "28a46762-6ac8-6cc4-4f55-bc9ab99af928",
  "detail-type": "Inspector2 Scan",
  "source": "aws.inspector2",
  "account": "111122223333",
  "time": "2023-01-20T22:52:35Z",
  "region": "us-east-1",
  "resources": [
    "i-087d63509b8c97098"
  ],
  "detail": {
    "scan-status": "INITIAL_SCAN_COMPLETE",
    "finding-severity-counts": {
      "CRITICAL": 0,
      "HIGH": 0,
      "MEDIUM": 0,
      "TOTAL": 0
    },
    "instance-id": "i-087d63509b8c97098",
    "version": "1.0"
  }
}
```

Amazon ECR image initial scan

```
{
  "version": "0",
  "id": "fdaa751a-984c-a709-44f9-9a9da9cd3606",
  "detail-type": "Inspector2 Scan",
  "source": "aws.inspector2",
  "account": "111122223333",
```

```

    "time": "2023-01-20T23:15:18Z",
    "region": "us-east-1",
    "resources": [
      "arn:aws:ecr:us-east-1:111122223333:repository/inspector2"
    ],
    "detail": {
      "scan-status": "INITIAL_SCAN_COMPLETE",
      "repository-name": "arn:aws:ecr:us-east-1:111122223333:repository/
inspector2",
      "finding-severity-counts": {
        "CRITICAL": 0,
        "HIGH": 0,
        "MEDIUM": 0,
        "TOTAL": 0
      },
      "image-digest":
"sha256:965fbcae990b0467ed5657caceaec165018ef44a4d2d46c7cdea80a9dff0d1ea",
      "image-tags": [
        "ubuntu22"
      ],
      "version": "1.0"
    }
  }
}

```

Lambda function initial scan

```

{
  "version": "0",
  "id": "4f290a7c-361b-c442-03c8-a629f6f20d6c",
  "detail-type": "Inspector2 Scan",
  "source": "aws.inspector2",
  "account": "111122223333",
  "time": "2023-02-23T18:06:03Z",
  "region": "us-west-2",
  "resources": [
    "arn:aws:lambda:us-west-2:111122223333:function:lambda-example:$LATEST"
  ],
  "detail": {
    "scan-status": "INITIAL_SCAN_COMPLETE",
    "finding-severity-counts": {

```

```
    "CRITICAL": 0,  
    "HIGH": 0,  
    "MEDIUM": 0,  
    "TOTAL": 0  
  },  
  "version": "1.0"  
}
```

Amazon Inspector coverage event schema example

The following is an example of the EventBridge event schema for an Amazon Inspector event for coverage. This event is created when Amazon Inspector scan coverage for a resource is changed. The following fields identify a coverage event:

- The `detail-type` field is set to `Inspector2 Coverage`.
- The `detail` object contains a `scanStatus` object that indicates the new scanning status for the resource.

```
{  
  "version": "0",  
  "id": "000adda5-0fbf-913e-bc0e-10f0376412aa",  
  "detail-type": "Inspector2 Coverage",  
  "source": "aws.inspector2",  
  "account": "111122223333",  
  "time": "2023-01-20T22:51:39Z",  
  "region": "us-east-1",  
  "resources": [  
    "i-087d63509b8c97098"  
  ],  
  "detail": {  
    "scanStatus": {  
      "reason": "UNMANAGED_EC2_INSTANCE",  
      "statusCodeValue": "INACTIVE"  
    },  
    "scanType": "PACKAGE",  
    "eventTimestamp": "2023-01-20T22:51:35.665501Z",
```

```
    "version": "1.0"  
  }  
}
```


Integrating Amazon Inspector scans into your CI/CD pipeline

The Amazon Inspector CI/CD integration utilizes the Amazon Inspector SBOM Generator and Amazon Inspector Scan API to produce vulnerability reports for container images. The Amazon Inspector SBOM Generator creates a software bill of materials (SBOM) for archives, container images, directories, local systems, and compiled Go and Rust binaries. The Amazon Inspector Scan API scans the SBOM to create a report with details about detected vulnerabilities. You can integrate Amazon Inspector container image scans with your CI/CD pipeline to scan for software vulnerabilities and produce vulnerability reports, which allow you to investigate and remediate risks before deployment. To set up your CI/CD integration, you can use plugins or create a custom CI/CD integration using the Amazon Inspector SBOM Generator and Amazon Inspector Scan API.

Topics

- [Plugin integration](#)
- [Custom integration](#)
- [Setting up an AWS account to use the Amazon Inspector CI/CD integration](#)
- [Amazon Inspector SBOM Generator](#)
- [Amazon Inspector Dockerfile checks](#)
- [Creating a custom CI/CD pipeline integration with Amazon Inspector Scan](#)
- [Using the Amazon Inspector Jenkins plugin](#)
- [Using the Amazon Inspector TeamCity plugin](#)
- [Using Amazon Inspector with GitHub actions](#)
- [Using Amazon Inspector with GitLab components](#)
- [Using CodeCatalyst actions with Amazon Inspector](#)
- [Using CycloneDX namespaces with Amazon Inspector](#)

Plugin integration

Amazon Inspector provides plugins for supported CI/CD solutions. You can install these plugins from their respective marketplaces and then use them to add Amazon Inspector Scans as a build

step in your pipeline. The plugin build step runs the Amazon Inspector SBOM generator on the image you supply, and then runs the Amazon Inspector Scan API on the generated SBOM.

The following is an overview of how an Amazon Inspector CI/CD integration works through plugins:

1. You configure an AWS account to allow access to the Amazon Inspector Scan API. For instructions, see [Setting up an AWS account to use the Amazon Inspector CI/CD integration](#).
2. You install the Amazon Inspector plugin from the marketplace.
3. You install and configure the Amazon Inspector SBOM Generator binary. For instructions, see [Amazon Inspector SBOM Generator](#).
4. You add Amazon Inspector Scans as a build step in your CI/CD pipeline and configure the scan.
5. When you run a build, the plugin takes your container image as input and then runs the Amazon Inspector SBOM Generator on the image to generate a CycloneDX compatible SBOM.
6. From there, the plugin sends the generated SBOM to an Amazon Inspector Scan API endpoint which assesses each SBOM component for vulnerabilities.
7. The Amazon Inspector Scan API response is transformed into a vulnerability report in CSV, SBOM JSON, and HTML formats. The report contains details about any vulnerabilities that Amazon Inspector found.

Supported CI/CD solutions

Amazon Inspector currently supports the following CI/CD solutions. For complete instructions on setting up the CI/CD integration using a plugin, select the plugin for your CI/CD solution:

- [Jenkins plugin](#)
- [TeamCity plugin](#)
- [GitHub actions](#)

Custom integration

If Amazon Inspector does not provide plugins for your CI/CD solution, you can create your own custom CI/CD integration using a combination of the Amazon Inspector SBOM Generator and the Amazon Inspector Scan API. You can also use a custom integration to fine-tune scans using the options available through Amazon Inspector SBOM Generator.

The following is an overview of how a custom Amazon Inspector CI/CD integration works:

1. You configure an AWS account to allow access to the Amazon Inspector Scan API. For instructions, see [Setting up an AWS account to use the Amazon Inspector CI/CD integration](#).
2. You install and configure the Amazon Inspector SBOM Generator binary. For instructions, see [Amazon Inspector SBOM Generator](#).
3. You use the Amazon Inspector SBOM Generator to generate a CycloneDX compatible SBOM for your container image.
4. You use the Amazon Inspector Scan API on the generated SBOM to produce a vulnerability report.

For instructions on setting up a custom integration, see [Creating a custom CI/CD pipeline integration with Amazon Inspector Scan](#).

Setting up an AWS account to use the Amazon Inspector CI/CD integration

To use the Amazon Inspector CI/CD integration, you must sign up for an AWS account. The AWS account must have an IAM role that grants your CI/CD pipeline access to the Amazon Inspector Scan API. Complete the tasks in the following topics to sign up for an AWS account, create an administrator user, and configure an IAM role for CI/CD integration.

Note

If you already signed up for an AWS account, you can skip to [Configure an IAM role for CI/CD integration](#).

Topics

- [Sign up for an AWS account](#)
- [Create a user with administrative access](#)
- [Configure an IAM role for CI/CD integration](#)

Sign up for an AWS account

If you do not have an AWS account, complete the following steps to create one.

To sign up for an AWS account

1. Open <https://portal.aws.amazon.com/billing/signup>.
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.

When you sign up for an AWS account, an *AWS account root user* is created. The root user has access to all AWS services and resources in the account. As a security best practice, assign administrative access to a user, and use only the root user to perform [tasks that require root user access](#).

AWS sends you a confirmation email after the sign-up process is complete. At any time, you can view your current account activity and manage your account by going to <https://aws.amazon.com/> and choosing **My Account**.

Create a user with administrative access

After you sign up for an AWS account, secure your AWS account root user, enable AWS IAM Identity Center, and create an administrative user so that you don't use the root user for everyday tasks.

Secure your AWS account root user

1. Sign in to the [AWS Management Console](#) as the account owner by choosing **Root user** and entering your AWS account email address. On the next page, enter your password.

For help signing in by using root user, see [Signing in as the root user](#) in the *AWS Sign-In User Guide*.

2. Turn on multi-factor authentication (MFA) for your root user.

For instructions, see [Enable a virtual MFA device for your AWS account root user \(console\)](#) in the *IAM User Guide*.

Create a user with administrative access

1. Enable IAM Identity Center.

For instructions, see [Enabling AWS IAM Identity Center](#) in the *AWS IAM Identity Center User Guide*.

2. In IAM Identity Center, grant administrative access to a user.

For a tutorial about using the IAM Identity Center directory as your identity source, see [Configure user access with the default IAM Identity Center directory](#) in the *AWS IAM Identity Center User Guide*.

Sign in as the user with administrative access

- To sign in with your IAM Identity Center user, use the sign-in URL that was sent to your email address when you created the IAM Identity Center user.

For help signing in using an IAM Identity Center user, see [Signing in to the AWS access portal](#) in the *AWS Sign-In User Guide*.

Assign access to additional users

1. In IAM Identity Center, create a permission set that follows the best practice of applying least-privilege permissions.

For instructions, see [Create a permission set](#) in the *AWS IAM Identity Center User Guide*.

2. Assign users to a group, and then assign single sign-on access to the group.

For instructions, see [Add groups](#) in the *AWS IAM Identity Center User Guide*.

Configure an IAM role for CI/CD integration

To integrate Amazon Inspector scanning into your CI/CD pipeline you need to create an IAM policy that allows access to the Amazon Inspector Scan API that scans the software bill of materials (SBOMs). Then, you can attach that policy to an IAM role that your account can assume to run the Amazon Inspector Scan API.

1. Sign in to the AWS Management Console and open the IAM console at <https://console.aws.amazon.com/iam/>.
2. In the navigation pane of the IAM console, **Policies** and then choose **Create Policy**.
3. In **Policy Editor** select **JSON** and paste the following statement:

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "VisualEditor0",
      "Effect": "Allow",
      "Action": "inspector-scan:ScanSbom",
      "Resource": "*"
    }
  ]
}
```

4. Choose **Next**.
5. Give the policy a name, for example `InspectorCICDscan-policy`, and add an optional description, then choose **Create Policy**. This policy will be attached to the role you'll create in the next steps.
6. In the navigation pane of the IAM console, select **Roles** and then select **Create New Role**.
7. For **Trusted entity type** choose **Custom trust policy** and paste the following policy:

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "AWS": "arn:aws:iam::{ACCOUNT_ID}:root"
      },
      "Action": "sts:AssumeRole",
      "Condition": {}
    }
  ]
}
```

8. Choose **Next**.
9. In **Add permissions** search for and select the policy you created earlier, then choose **Next**.
10. Give the role a name, for example `InspectorCICDscan-role`, and add an optional description, then choose **Create Role**.

Amazon Inspector SBOM Generator

A Software Bill of Materials (SBOM) is [a formally structured list of components, libraries, and modules](#) required to build a piece of software. The Amazon Inspector SBOM Generator (Sbomgen) is a tool that produces an SBOM for archives, container images, directories, local systems, and compiled Go and Rust binaries. Sbomgen scans for files that contain information about installed packages. When Sbomgen finds a relevant file, it extracts package names, versions, and other metadata. Sbomgen then transforms package metadata into a CycloneDX SBOM. You can use Sbomgen to generate the CycloneDX SBOM as a file or in STDOUT and send SBOMs to Amazon Inspector for vulnerability detection. You can also use Sbomgen as part of [the CI/CD integration](#), which scans container images automatically as part of your deployment pipeline.

Supported packages types

Sbomgen collects inventory for the following package types:

- Alpine APK
- Debian/Ubuntu DPKG
- Red Hat RPM
- C#
- Go
- Java
- Node.js
- PHP
- Python
- Ruby
- Rust

Supported container image configuration checks

Sbomgen can scan standalone Dockerfiles and build history from existing images for security issues. For more information, see [Amazon Inspector Dockerfile checks](#).

Installing Sbomgen

Sbomgen is only available for Linux operating systems.

You must have Docker installed if you want Sbomgen to analyze locally cached images. Docker isn't required to analyze images exported as `.tar` files or images hosted in remote container registries.

Amazon Inspector recommends that you run Sbomgen from a system with at least the following hardware specs:

- 4x core CPU
- 8 GB RAM

To install Sbomgen

1. Download the latest Sbomgen zip file from the correct URL for your architecture:

Linux AMD64: <https://amazon-inspector-sbomgen.s3.amazonaws.com/latest/linux/amd64/inspector-sbomgen.zip>

Linux ARM64: <https://amazon-inspector-sbomgen.s3.amazonaws.com/latest/linux/arm64/inspector-sbomgen.zip>

Alternatively, you can download [previous versions of the Amazon Inspector SBOM Generator zip file](#).

2. Unzip the download using the following command:

```
unzip inspector-sbomgen.zip
```

3. Check for the following files in the extracted directory:

- `inspector-sbomgen` – This is the tool you will execute to generate SBOMs.
- `README.txt` – This is the documentation for using Sbomgen.
- `LICENSE.txt` – This file contains the software license for Sbomgen.

- `licenses` – This folder contains license info for third party packages used by Sbmngen.
 - `checksums.txt` – This file provides hashes of the Sbmngen tool.
 - `sbom.json` – This is a CycloneDX SBOM for the Sbmngen tool.
 - `WhatsNew.txt` – This file contains a summarized change log, so you can view major changes and improvements between Sbmngen versions quickly.
4. (Optional) Verify the authenticity and integrity of the tool using the following command:

```
sha256sum < inspector-sbmngen
```

- Compare the results with the contents of the `checksums.txt` file.
5. Grant executable permissions to the tool using the following command:

```
chmod +x inspector-sbmngen
```

6. Verify that Sbmngen is successfully installed using the following command:

```
./inspector-sbmngen --version
```

You should see the output similar to the following:

```
Version: 1.X.X
```

Using Sbmngen

This section describes different ways you can use Sbmngen. You can learn more about how to use Sbmngen through built-in examples. To view these examples, run the `list-examples` command:

```
./inspector-sbmngen list-examples
```

Generate an SBOM for a container image and output the result

You can use Sbmngen to generate SBOMs for container images and output the result to a file. This capability can be enabled using the `container` subcommand.

Example command

In the following snippet, you can replace `image:tag` with the ID of your image and `output_path.json` with the path to the output you want to save.

```
# generate SBOM for container image
./inspector-sbomgen container --image image:tag -o output_path.json
```

Note

Scan time and performance depends on the image size and how small the number of layers are. Smaller images not only improve Sbmngen performance, but also reduce the potential attack surface. Smaller images also improve image build, download, and upload times.

When using Sbmngen with [ScanSbom](#), the Amazon Inspector Scan API won't process SBOMs that contain more than 2,000 packages. In this scenario, the Amazon Inspector Scan API returns an HTTP 400 response.

If an image includes bulk media files or directories, consider excluding them from Sbmngen using the `--skip-files` argument.

Generate an SBOM from directories and archives

You can use Sbmngen to generate SBOMs from directories and archives. This capability can be enabled using the `directory` or `archive` subcommands. Amazon Inspector recommends using this feature when you want to generate an SBOM from a project folder, such as a downloaded git repository.

Example command 1

The following snippet shows a subcommand that generates an SBOM from a directory file.

```
# generate SBOM from directory
./inspector-sbomgen directory --path /path/to/dir -o /tmp/sbom.json
```

Example command 2

The following snippet shows a subcommand that generates an SBOM from an archive file. The only supported archive formats are `.zip`, `.tar`, and `.tar.gz`.

```
# generate SBOM from archive file (tar, tar.gz, and zip formats only)
./inspector-sbomgen archive --path testData.zip -o /tmp/sbom.json
```

Generate an SBOM from Go or Rust compiled binaries

You can use Sbmngen to generate SBOMs from compiled Go and Rust binaries. You can enable this capability through the `binary` subcommand:

```
./inspector-sbmngen binary --path /path/to/your/binary
```

Send an SBOM to Amazon Inspector for vulnerability identification

In addition to generating an SBOM, you can send an SBOM for scanning with a single command from the Amazon Inspector Scan API. Amazon Inspector evaluates the contents of the SBOM for vulnerabilities before returning findings to Sbmngen. Depending on your input, the findings can be displayed or written to a file.

Note

You must have an active AWS account with read permissions to `InspectorScan-ScanSbom` to use this capability.

To enable this capability, you pass the `--scan-sbom` argument to the Sbmngen CLI. You can also pass the `--scan-sbom` argument to any of the following Sbmngen subcommands: `archive`, `binary`, `container`, `directory`, `localhost`.

Note

The Amazon Inspector Scan API doesn't process SBOMs with more than 2,000 packages. In this scenario, the Amazon Inspector Scan API returns an HTTP 400 response.

You can authenticate to Amazon Inspector through an AWS profile or an IAM role with the following AWS CLI arguments:

```
--aws-profile profile  
--aws-region region  
--aws-iam-role-arn role_arn
```

You can also authenticate to Amazon Inspector by providing the following environment variables to Sbmongen.

```
AWS_ACCESS_KEY_ID=$access_key \  
AWS_SECRET_ACCESS_KEY=$secret_key \  
AWS_DEFAULT_REGION=$region \  
./inspector-sbmongen arguments
```

To specify the response format, use the `--scan-sbom-output-format cyclonedx` argument or `--scan-sbom-output-format inspector` argument.

Example command 1

This command creates an SBOM for the latest Alpine Linux release, scans the SBOM, and writes the vulnerability results to a JSON file.

```
./inspector-sbmongen container --image alpine:latest \  
    --scan-sbom \  
    --aws-profile your_profile \  
    --aws-region your_region \  
    --scan-sbom-output-format cyclonedx \  
    --outfile /tmp/inspector_scan.json
```

Example command 2

This command authenticates to Amazon Inspector using AWS credentials as environment variables.

```
AWS_ACCESS_KEY_ID=$your_access_key \  
AWS_SECRET_ACCESS_KEY=$your_secret_key \  
AWS_DEFAULT_REGION=$your_region \  
./inspector-sbmongen container --image alpine:latest \  
    -o /tmp/sbom.json \  
    --scan-sbom \  
    --scan-sbom-output-format inspector
```

Example command 3

This command authenticates to Amazon Inspector using the ARN for an IAM role.

```
./inspector-sbomgen container --image alpine:latest \  
    --scan-sbom \  
    --aws-profile your_profile \  
    --aws-region your_region \  
    --outfile /tmp/inspector_scan.json  
    --aws-iam-role-arn arn:aws:iam::123456789012:role/your_role
```

Customize scans to exclude specific files

When analyzing and processing a container image, Sbomgen scans the size of all files in that container image. You can customize scans to exclude specific files or target specific packages.

To reduce disk consumption, RAM consumption, elapsed runtime, and skip files that exceed the provided threshold, use the `--max-file-size` argument with the `container` subcommand:

```
./inspector-sbomgen container --image alpine:latest \  
--outfile /tmp/sbom.json \  
--max-file-size 300000000
```

Disable progress indicator

Sbomgen displays a spinning progress indicator that can result in excessive slash characters in CI/CD environments.

```
INFO[2024-02-01 14:58:46]coreV1.go:53: analyzing artifact  
|  
\  
/  
|  
\  
/  
INFO[2024-02-01 14:58:46]coreV1.go:62: executing post-processors
```

You can disable the progress indicator using the `--disable-progress-bar` argument:

```
./inspector-sbomgen container --image alpine:latest \  
--outfile /tmp/sbom.json \  
--disable-progress-bar
```

Authenticating to private registries with Sbomgen

By providing your private registry authentication credentials, you can generate SBOMs from containers that are hosted in private registries. You can provide these credentials through the following methods:

Authenticate using cached credentials (recommended)

For this method, you authenticate to your container registry. For example, if using Docker, you can authenticate to your container registry using the Docker logging command: `docker login`.

1. Authenticate to your container registry. For example, if using Docker, you can authenticate to your registry using the Docker `login` command:
2. After you authenticate to your container registry, use Sbomgen on a container image that's in the registry. To use the following example, replace *image:tag* with the name of the image to scan:

```
./inspector-sbomgen container --image image:tag
```

Authenticate using the interactive method

For this method, provide your username as a parameter, and Sbomgen will prompt you for secure password entry when needed.

To use the following example, replace *image:tag* with the name of the image that you want to scan and *your_username* with a username that has access to the image:

```
./inspector-sbomgen container --image image:tag --username your_username
```

Authenticate using the non-interactive method

For this method, store your password or registry token in a `.txt` file.

Note

The current user should only be able to read this file. The file should also contain your password or token on a single line.

To use the following example, replace *your_username* with your username, *password.txt* with the .txt file that includes your password or token on a single line, and *image:tag* with the name of the image to scan:

```
INSPECTOR_SBOMGEN_USERNAME=your_username \  
INSPECTOR_SBOMGEN_PASSWORD=`cat password.txt` \  
./inspector-sbomgen container --image image:tag
```

Example outputs from Sbomgen

The following is an example of an SBOM for a container image inventoried using Sbomgen.

Container image SBOM

```
{  
  "bomFormat": "CycloneDX",  
  "specVersion": "1.5",  
  "serialNumber": "urn:uuid:828875ef-8c32-4777-b688-0af96f3cf619",  
  "version": 1,  

```

```

      "value":
"sha256:c81c8ae4dda7dedc0711daefe4076d33a88a69a28c398688090c1141eff17e50"
    },
    {
      "name": "amazon:inspector:sbom_generator:layer_diff_id",
      "value":
"sha256:eddd0d48c295dc168d0710f70364581bd84b1dda6bb386c4a4de0b61de2f2119"
    }
  ]
}
},
"components": [
  {
    "bom-ref": "comp-2",
    "type": "library",
    "name": "dnf",
    "version": "4.18.0",
    "purl": "pkg:pypi/dnf@4.18.0",
    "properties": [
      {
        "name": "amazon:inspector:sbom_generator:source_file_scanner",
        "value": "python-pkg"
      },
      {
        "name": "amazon:inspector:sbom_generator:source_package_collector",
        "value": "python-pkg"
      },
      {
        "name": "amazon:inspector:sbom_generator:source_path",
        "value": "/usr/lib/python3.12/site-packages/dnf-4.18.0.dist-info/METADATA"
      },
      {
        "name": "amazon:inspector:sbom_generator:is_duplicate_package",
        "value": "true"
      },
      {
        "name": "amazon:inspector:sbom_generator:duplicate_purl",
        "value": "pkg:rpm/fedora/python3-dnf@4.18.0-2.fc39?
arch=noarch&distro=39&epoch=0"
      }
    ]
  },
  {
    "bom-ref": "comp-3",

```



```

    "type": "library",
    "name": "libcomps",
    "version": "0.1.20",
    "purl": "pkg:pypi/libcomps@0.1.20",
    "properties": [
      {
        "name": "amazon:inspector:sbom_generator:source_file_scanner",
        "value": "python-pkg"
      },
      {
        "name": "amazon:inspector:sbom_generator:source_package_collector",
        "value": "python-pkg"
      },
      {
        "name": "amazon:inspector:sbom_generator:source_path",
        "value": "/usr/lib64/python3.12/site-packages/libcomps-0.1.20-py3.12.egg-
info/PKG-INFO"
      },
      {
        "name": "amazon:inspector:sbom_generator:is_duplicate_package",
        "value": "true"
      },
      {
        "name": "amazon:inspector:sbom_generator:duplicate_purl",
        "value": "pkg:rpm/fedora/python3-libcomps@0.1.20-1.fc39?
arch=x86_64&distro=39&epoch=0"
      }
    ]
  }
]
}

```

Previous versions of the Amazon Inspector SBOM Generator

This topic provides you with links to previous versions of the Amazon Inspector SBOM Generator. For information about installing Sbmngen, see [Installing Sbmngen](#).

Sbmngen 1.5.0

- Linux AMD64: <https://amazon-inspector-sbomgen.s3.amazonaws.com/1.5.0/linux/amd64/inspector-sbomgen.zip>

- Linux ARM64: <https://amazon-inspector-sbomgen.s3.amazonaws.com/1.5.0/linux/arm64/inspector-sbomgen.zip>

Sbomgen 1.4.0

- Linux AMD64: <https://amazon-inspector-sbomgen.s3.amazonaws.com/1.4.0/linux/amd64/inspector-sbomgen.zip>
- Linux ARM64: <https://amazon-inspector-sbomgen.s3.amazonaws.com/1.4.0/linux/arm64/inspector-sbomgen.zip>

Sbomgen 1.3.2

- Linux AMD64: <https://amazon-inspector-sbomgen.s3.amazonaws.com/1.3.2/linux/amd64/inspector-sbomgen.zip>
- Linux ARM64: <https://amazon-inspector-sbomgen.s3.amazonaws.com/1.3.2/linux/arm64/inspector-sbomgen.zip>

Sbomgen 1.3.1

- Linux AMD64: <https://amazon-inspector-sbomgen.s3.amazonaws.com/1.3.1/linux/amd64/inspector-sbomgen.zip>
- Linux ARM64: <https://amazon-inspector-sbomgen.s3.amazonaws.com/1.3.1/linux/arm64/inspector-sbomgen.zip>

Sbomgen 1.3.0

- Linux AMD64: <https://amazon-inspector-sbomgen.s3.amazonaws.com/1.3.0/linux/amd64/inspector-sbomgen.zip>
- Linux ARM64: <https://amazon-inspector-sbomgen.s3.amazonaws.com/1.3.0/linux/arm64/inspector-sbomgen.zip>

Sbomgen 1.2.1

- Linux AMD64: <https://amazon-inspector-sbomgen.s3.amazonaws.com/1.2.1/linux/amd64/inspector-sbomgen.zip>

- Linux ARM64: <https://amazon-inspector-sbomgen.s3.amazonaws.com/1.2.1/linux/arm64/inspector-sbomgen.zip>

Sbomgen 1.2.0

- Linux AMD64: <https://amazon-inspector-sbomgen.s3.amazonaws.com/1.2.0/linux/amd64/inspector-sbomgen.zip>
- Linux ARM64: <https://amazon-inspector-sbomgen.s3.amazonaws.com/1.2.0/linux/arm64/inspector-sbomgen.zip>

Sbomgen 1.1.1

- Linux AMD64: <https://amazon-inspector-sbomgen.s3.amazonaws.com/1.1.1/linux/amd64/inspector-sbomgen.zip>
- Linux ARM64: <https://amazon-inspector-sbomgen.s3.amazonaws.com/1.1.1/linux/arm64/inspector-sbomgen.zip>

Sbomgen 1.1.0

- Linux AMD64: <https://amazon-inspector-sbomgen.s3.amazonaws.com/1.1.0/linux/amd64/inspector-sbomgen.zip>
- Linux ARM64: <https://amazon-inspector-sbomgen.s3.amazonaws.com/1.1.0/linux/arm64/inspector-sbomgen.zip>

Sbomgen 1.0.0

- Linux AMD64: <https://amazon-inspector-sbomgen.s3.amazonaws.com/1.0.0/linux/amd64/inspector-sbomgen.zip>
- Linux ARM64: <https://amazon-inspector-sbomgen.s3.amazonaws.com/1.0.0/linux/arm64/inspector-sbomgen.zip>

Amazon Inspector Dockerfile checks

This section describes how to use the Amazon Inspector SBOM Generator to scan Dockerfiles and Docker container images for misconfigurations that introduce security vulnerabilities.

Topics

- [Using Sbomgen Dockerfile checks](#)
- [Supported Dockerfile checks](#)

Using Sbomgen Dockerfile checks

Dockerfile checks are conducted automatically when a file named Dockerfile or *.Dockerfile is discovered and when a Docker image is scanned.

You can disable Dockerfile checks using the `--skip-scanners dockerfile` argument. You also can combine Dockerfile checks with any available scanner, such as OS or 3rd-party packages.

Example Docker check commands

The following example commands show how to generate SBOMs for Dockerfiles and Docker container images, as well as for OS and 3rd-party packages.

```
# generate SBOM only containing Docker checks for Dockerfiles in a local directory
./inspector-sbomgen directory --path ./project/ --scanners dockerfile

# generate SBOM for container image will by default include Dockerfile checks
./inspector-sbomgen container --image image:tag

# generate SBOM only containing Docker checks for specific Dockerfiles and Alpine,
  Debian, and RHEL OS packages in a local directory
./inspector-sbomgen directory --path ./project/ --scanners dockerfile,dpkg,alpine-
apk,rhel-rpm

# generate SBOM only containing Docker checks for specific Dockerfiles in a local
  directory
./inspector-sbomgen directory --path ./project/ --skip-scanners dockerfile
```

Example file component

The following is an example of a Dockerfile finding for a file component.

```
{
  "bom-ref": "comp-2",
  "name": "dockerfile:data/docker/Dockerfile",
  "properties": [
```

```

    {
      "name": "amazon:inspector:sbom_scanner:dockerfile_finding:IN-DOCKER-001",
      "value": "affected_lines:27-27"
    }
  ],
  "type": "file"
},

```

Example vulnerability response component

The following is an example of a Dockerfile finding for a vulnerability response component.

```

{
  "advisories": [
    {
      "url": "https://docs.docker.com/develop/develop-images/instructions/"
    }
  ],
  "affects": [
    {
      "ref": "comp-2"
    }
  ],
  "analysis": {
    "state": "in_triage"
  },
  "bom-ref": "vuln-13",
  "created": "2024-03-27T14:36:39Z",
  "description": "apt-get layer caching: Using apt-get update alone in a RUN statement causes caching issues and subsequent apt-get install instructions to fail.",
  "id": "IN-DOCKER-001",
  "ratings": [
    {
      "method": "other",
      "severity": "info",
      "source": {
        "name": "AMAZON_INSPECTOR",
        "url": "https://aws.amazon.com/inspector/"
      }
    }
  ],
  "source": {
    "name": "AMAZON_INSPECTOR",
    "url": "https://aws.amazon.com/inspector/"
  }
}

```

```
  },  
  "updated": "2024-03-27T14:36:39Z"  
},
```

Note

If you invoke Sbmngen without the `--scan-sbom` flag, you can only view raw Dockerfile findings.

Supported Dockerfile checks

Sbmngen Dockerfile checks are supported for the following:

- The Sudo binary package
- Debian APT utilities
- Hardcoded secrets
- Root containers
- Runtime weakening command flags
- Runtime weakening environment variables

Each of these Dockerfile checks has a corresponding severity rating, which is noted at the top of the following topics.

Note

The recommendations described in the following topics are based on industry best practices.

The Sudo binary package

Note

The severity rating for this check is **Info**.

We recommend not installing or using the Sudo binary package because it has unpredictable TTY and signal-forwarding behavior. For more information, see [User](#) in the Docker Docs website. If your use case requires functionality similar the Sudo binary package, we recommended using [Gosu](#).

Debian APT utilities

Note

The severity rating for this check is **High**.

The following are best practices for using Debian APT utilities.

Combining apt-get commands in a single Run statement to avoid caching issues

We recommend combining apt-get commands in a single RUN statement inside of your Docker container. Using apt-get update by itself results in caching issues and subsequent apt-get install instructions to fail. For more information, see [apt-get](#) in the Docker Docs website.

Note

The caching behavior described also can occur inside of your Docker container if the Docker container software is out of date.

Using the APT command-line utility in a non-interactive manner

We recommend using the APT command-line utility interactively. The APT command-line utility is designed as an end-user tool, and its behavior changes between versions. For more information, see [Script Usage and differences from other APT tools](#) in the Debian website.

Hard-coded secrets

Note

The severity rating for this check is **Critical**.

Confidential information in your Dockerfile is considered a hard-coded secret. The following hard-coded secrets can be identified through Sbomgen Docker file checks:

- AWS access key IDs – AKIAIOSFODNN7EXAMPLE
- DockerHub personal access tokens – dckr_pat_thisisa27charexample1234567
- GitHub personal access tokens – ghp_examplev61wY7Pj1YnotrealUoY123456789
- GitLab personal access tokens – glpat-12345example12345678

Root containers

Note

The severity marker for this check is **Info**.

We recommend running Docker containers without root privileges. For containerized workloads that cannot run without root privileges, we recommend building your applications using a principle with the least amount of privileges. For more information, see [User](#) in the Docker Docs website.

Runtime weakening environment variables

Note

The severity rating for this check is **High**.

Several command line utilities or programming language runtimes support bypassing secure defaults, which allows execution through insecure methods.

NODE_TLS_REJECT_UNAUTHORIZED=0

When Node.js processes run with `NODE_TLS_REJECT_UNAUTHORIZED` set to `0`, TLS certificate validation is disabled. For more information, see [NODE_TLS_REJECT_UNAUTHORIZED=0](#) in the Node.js website.

GIT_SSL_NO_VERIFY=*

When git command line processes run with `GIT_SSL_NO_VERIFY` set, Git skips verifying TLS certificates. For more information, see [Environment variables](#) in the Git website.

PIP_TRUSTED_HOST=*

When Python pip command line processes run with PIP_TRUSTED_HOST set, Pip skips verifying TLS certificates on the specified domain. For more information, see [--trusted-host](#) in the Pip website.

NPM_CONFIG_STRICT_SSL=false

When Node.js npm command line processes run with NPM_CONFIG_STRICT_SSL set to false, the Node Package Manager (npm) utility will connect to the NPM registry without validating TLS certificates. For more information, see [strict-ssl](#) in the npm Docs website.

Runtime weakening command flags

Note

The severity rating for this check is **High**.

Similar to runtime weakening environment variables, several command line utilities or programming language runtimes support bypassing secure defaults, which allows execution through insecure methods.

npm --strict-ssl=false

When Node.js npm command line processes are run with the `--strict-ssl=false` flag, the Node Package Manager (npm) utility connects to the NPM registry without validating TLS certificates. For more information, see [strict-ssl](#) in the npm Docs website.

apk --allow-untrusted

When the Alpine Package Keeper utility is run with the `--allow-untrusted` flag, apk will install packages with no or untrusted signatures. For more information, see [the following repository](#) in the Apline website.

apt-get --allow-unauthenticated

When the Debian apt-get package utility is run with the `--allow-unauthenticated` flag, apt-get doesn't check package validity. For more information, see [APT-Get\(8\)](#) in the Debian website.

pip --trusted-host

When the Python pip utility is run with the `--trusted-host` flag, the specified hostname will bypass TLS certificate validation. For more information, see [--trusted-host](#) in the Pip website.

rpm --nodigest, --nosignature, --noverify, --nofiledigest

When the RPM-based package manager `rpm` is run with the `--nodigest`, `--nosignature`, `--noverify`, and `--nofiledigest` flags, the RPM package manager doesn't validate package headers, signatures, or files when installing a package. For more information, see the following [RPM manual page](#) in the RPM website.

yum-config-manager --setopt=sslverify false

When the RPM-based package manager `yum-config-manager` is run with the `--setopt=sslverify` flag set to `false`, the YUM package manager doesn't validate TLS certificates. For more information, see the following [YUM manual page](#) in the Man7 website.

yum --nogpgcheck

When the RPM-based package manager `yum` is run with the `--nogpgcheck` flag, the YUM package manager skips checking GPG signatures on packages. For more information, see [yum\(8\)](#) in the Man7 website.

curl --insecure, curl -k

When `curl` is run with the `--insecure` or `-k` flag, TLS certificate validation is disabled. By default, every secure connection that `curl` makes is verified to be secure before the transfer takes place. This option makes `curl` skip the verification step and proceed without checking. For more information, see the following [Curl manual page](#) in the Curl website.

wget --no-check-certificate

When `wget` is run with the `--no-check-certificate` flag, TLS certificate validation is disabled. For more information, see the following [Wget manual page](#) in the GNU website.

Creating a custom CI/CD pipeline integration with Amazon Inspector Scan

We recommend that you use the [Amazon Inspector CI/CD plugins](#) if the Amazon Inspector CI/CD plugins are available for your CI/CD solution. If the Amazon Inspector CI/CD plugins aren't available for your CI/CD solution, you can use a combination of the Amazon Inspector SBOM Generator and the Amazon Inspector Scan API to create a custom CI/CD integration. The following steps describe how to create a custom CI/CD pipeline integration with Amazon Inspector Scan.

Tip

You can use the [Amazon Inspector SBOM Generator \(Sbomgen\)](#) to skip Step 3 and Step 4 if you want to [generate and scan your SBOM in a single command](#).

Step 1. Configuring AWS account

Configure an AWS account that provides access to the Amazon Inspector Scan API. For more information, see [Setting up an AWS account to use the Amazon Inspector CI/CD integration](#).

Step 2. Installing Sbomgen binary

Install and configure the Sbomgen binary. For more information, see [Installing Sbomgen](#).

Step 3. Using Sbomgen

Use the Sbomgen to create an SBOM file for a container image that you want to scan.

You can use the following example. Replace *image:id* with the name of the image that you to scan. Replace *sbom_path.json* with the location where you want to save the SBOM output.

Example

```
./inspector-sbomgen container --image image:id -o sbom_path.json
```

Step 4. Calling the Amazon Inspector Scan API

Call the `inspector-scan` API to scan the generated SBOM and provide a vulnerability report.

You can use the following example. Replace *sbom_path.json* with the location of a valid CycloneDX compatible SBOM file. Replace *ENDPOINT* with the API endpoint for the AWS Region where you're currently authenticated. Replace *REGION* with the corresponding Region.

Example

```
aws inspector-scan scan-sbom --sbom file://sbom_path.json --endpoint ENDPOINT-URL --region REGION
```

For a complete list of AWS Regions and endpoints, see [Regions and endpoints](#).

(Optional) Step 5. Generate and scan SBOM in a single command

Note

Only complete this step if you skipped Step 3 and Step 4.

Generate and scan your SBOM in a single command using the `--scan-bom` flag.

You can use the following example. Replace `image:id` with the name of the image that you want to scan. Replace `profile` with the corresponding profile. Replace `REGION` with the corresponding Region. Replace `/tmp/scan.json` with the location of the scan.json file in the tmp directory.

Example

```
./inspector-sbomgen container --image image:id --scan-sbom --aws-profile profile --aws-region REGION -o /tmp/scan.json
```

For a complete list of AWS Regions and endpoints, see [Regions and endpoints](#).

API output formats

The Amazon Inspector Scan API can output a vulnerability report in CycloneDX 1.5 format or Amazon Inspector finding JSON. The default can be changed using the `--output-format` flag.

Example of CycloneDX 1.5 format output

```
{
  "status": "SBOM parsed successfully, 1 vulnerabilities found",
  "sbom": {
    "bomFormat": "CycloneDX",
    "specVersion": "1.5",
    "serialNumber": "urn:uuid:0077b45b-ff1e-4dbb-8950-ded11d8242b1",
    "metadata": {
      "properties": [
        {
          "name": "amazon:inspector:sbom_scanner:critical_vulnerabilities",
          "value": "1"
        },
        {
          "name": "amazon:inspector:sbom_scanner:high_vulnerabilities",
          "value": "0"
        }
      ]
    }
  }
}
```

```
    },
    {
      "name": "amazon:inspector:sbom_scanner:medium_vulnerabilities",
      "value": "0"
    },
    {
      "name": "amazon:inspector:sbom_scanner:low_vulnerabilities",
      "value": "0"
    }
  ],
  "tools": [
    {
      "name": "CycloneDX SBOM API",
      "vendor": "Amazon Inspector",
      "version": "empty:083c9b00:083c9b00:083c9b00"
    }
  ],
  "timestamp": "2023-06-28T14:15:53.760Z"
},
"components": [
  {
    "bom-ref": "comp-1",
    "type": "library",
    "name": "log4j-core",
    "purl": "pkg:maven/org.apache.logging.log4j/log4j-core@2.12.1",
    "properties": [
      {
        "name": "amazon:inspector:sbom_scanner:path",
        "value": "/home/dev/foo.jar"
      }
    ]
  }
],
"vulnerabilities": [
  {
    "bom-ref": "vuln-1",
    "id": "CVE-2021-44228",
    "source": {
      "name": "NVD",
      "url": "https://nvd.nist.gov/vuln/detail/CVE-2021-44228"
    },
    "references": [
      {
        "id": "SNYK-JAVA-ORGAPACHELOGGINGLOG4J-2314720",
```

```
    "source": {
      "name": "SNYK",
      "url": "https://security.snyk.io/vuln/SNYK-JAVA-
ORGAPACHELOGGINGLOG4J-2314720"
    }
  },
  {
    "id": "GHSA-jfh8-c2jp-5v3q",
    "source": {
      "name": "GITHUB",
      "url": "https://github.com/advisories/GHSA-jfh8-c2jp-5v3q"
    }
  }
],
"ratings": [
  {
    "source": {
      "name": "NVD",
      "url": "https://www.first.org/cvss/v3-1/"
    },
    "score": 10.0,
    "severity": "critical",
    "method": "CVSSv31",
    "vector": "AV:N/AC:L/PR:N/UI:N/S:C/C:H/I:H/A:H"
  },
  {
    "source": {
      "name": "NVD",
      "url": "https://www.first.org/cvss/v2/"
    },
    "score": 9.3,
    "severity": "critical",
    "method": "CVSSv2",
    "vector": "AC:M/Au:N/C:C/I:C/A:C"
  },
  {
    "source": {
      "name": "EPSS",
      "url": "https://www.first.org/epss/"
    },
    "score": 0.97565,
    "severity": "none",
    "method": "other",
    "vector": "model:v2023.03.01,date:2023-06-27T00:00:00+0000"
```

```

    },
    {
      "source": {
        "name": "SNYK",
        "url": "https://security.snyk.io/vuln/SNYK-JAVA-
ORGAPACHELOGGINGLOG4J-2314720"
      },
      "score": 10.0,
      "severity": "critical",
      "method": "CVSSv31",
      "vector": "AV:N/AC:L/PR:N/UI:N/S:C/C:H/I:H/A:H/E:H"
    },
    {
      "source": {
        "name": "GITHUB",
        "url": "https://github.com/advisories/GHSA-jfh8-c2jp-5v3q"
      },
      "score": 10.0,
      "severity": "critical",
      "method": "CVSSv31",
      "vector": "AV:N/AC:L/PR:N/UI:N/S:C/C:H/I:H/A:H"
    }
  ],
  "cwes": [
    400,
    20,
    502
  ],
  "description": "Apache Log4j2 2.0-beta9 through 2.15.0 (excluding security
releases 2.12.2, 2.12.3, and 2.3.1) JNDI features used in configuration, log messages,
and parameters do not protect against attacker controlled LDAP and other JNDI related
endpoints. An attacker who can control log messages or log message parameters can
execute arbitrary code loaded from LDAP servers when message lookup substitution is
enabled. From log4j 2.15.0, this behavior has been disabled by default. From version
2.16.0 (along with 2.12.2, 2.12.3, and 2.3.1), this functionality has been completely
removed. Note that this vulnerability is specific to log4j-core and does not affect
log4net, log4cxx, or other Apache Logging Services projects.",
  "advisories": [
    {
      "url": "https://www.intel.com/content/www/us/en/security-center/advisory/
intel-sa-00646.html"
    },
    {
      "url": "https://support.apple.com/kb/HT213189"
    }
  ]
}

```

```
    },
    {
      "url": "https://msrc-blog.microsoft.com/2021/12/11/microsofts-response-to-cve-2021-44228-apache-log4j2/"
    },
    {
      "url": "https://logging.apache.org/log4j/2.x/security.html"
    },
    {
      "url": "https://www.debian.org/security/2021/dsa-5020"
    },
    {
      "url": "https://cert-portal.siemens.com/productcert/pdf/ssa-479842.pdf"
    },
    {
      "url": "https://www.oracle.com/security-alerts/alert-cve-2021-44228.html"
    },
    {
      "url": "https://www.oracle.com/security-alerts/cpujan2022.html"
    },
    {
      "url": "https://cert-portal.siemens.com/productcert/pdf/ssa-714170.pdf"
    },
    {
      "url": "https://lists.fedoraproject.org/archives/list/package-announce@lists.fedoraproject.org/message/M5CSVUNV4HWZZXG0KNSK6L7RPM7B0KIB/"
    },
    {
      "url": "https://cert-portal.siemens.com/productcert/pdf/ssa-397453.pdf"
    },
    {
      "url": "https://cert-portal.siemens.com/productcert/pdf/ssa-661247.pdf"
    },
    {
      "url": "https://lists.fedoraproject.org/archives/list/package-announce@lists.fedoraproject.org/message/VU57UJDCFIASI035GC55JMKSRXJMCDFM/"
    },
    {
      "url": "https://www.oracle.com/security-alerts/cpuapr2022.html"
    },
    {
      "url": "https://twitter.com/kurtseifried/status/1469345530182455296"
    }
  ],
}
```



```

        "url": "https://tools.cisco.com/security/center/content/
CiscoSecurityAdvisory/cisco-sa-apache-log4j-qRuKNEbd"
    },
    {
        "url": "https://lists.debian.org/debian-lts-announce/2021/12/msg00007.html"
    },
    {
        "url": "https://www.kb.cert.org/vuls/id/930724"
    }
],
"created": "2021-12-10T10:15:00Z",
"updated": "2023-04-03T20:15:00Z",
"affects": [
    {
        "ref": "comp-1"
    }
],
"properties": [
    {
        "name": "amazon:inspector:sbom_scanner:exploit_available",
        "value": "true"
    },
    {
        "name": "amazon:inspector:sbom_scanner:exploit_last_seen_in_public",
        "value": "2023-03-06T00:00:00Z"
    },
    {
        "name": "amazon:inspector:sbom_scanner:cisa_kev_date_added",
        "value": "2021-12-10T00:00:00Z"
    },
    {
        "name": "amazon:inspector:sbom_scanner:cisa_kev_date_due",
        "value": "2021-12-24T00:00:00Z"
    },
    {
        "name": "amazon:inspector:sbom_scanner:fixed_version:comp-1",
        "value": "2.15.0"
    }
]
}
]
}
}

```

Example of Inspector format output

```
    {
"status": "SBOM parsed successfully, 1 vulnerability found",
"inspector": {
  "messages": [
    {
      "name": "foo",
      "purl": "pkg:maven/foo@1.0.0", // Will not exist in output if missing in sbom
      "info": "Component skipped: no rules found."
    }
  ],
  "vulnerability_count": {
    "critical": 1,
    "high": 0,
    "medium": 0,
    "low": 0
  },
  "vulnerabilities": [
    {
      "id": "CVE-2021-44228",
      "severity": "critical",
      "source": "https://nvd.nist.gov/vuln/detail/CVE-2021-44228",
      "related": [
        "SNYK-JAVA-ORGAPACHELOGGINGLOG4J-2314720",
        "GHSA-jfh8-c2jp-5v3q"
      ],
      "description": "Apache Log4j2 2.0-beta9 through 2.15.0 (excluding security releases 2.12.2, 2.12.3, and 2.3.1) JNDI features used in configuration, log messages, and parameters do not protect against attacker controlled LDAP and other JNDI related endpoints. An attacker who can control log messages or log message parameters can execute arbitrary code loaded from LDAP servers when message lookup substitution is enabled. From log4j 2.15.0, this behavior has been disabled by default. From version 2.16.0 (along with 2.12.2, 2.12.3, and 2.3.1), this functionality has been completely removed. Note that this vulnerability is specific to log4j-core and does not affect log4net, log4cxx, or other Apache Logging Services projects.",
      "references": [
        "https://www.intel.com/content/www/us/en/security-center/advisory/intel-sa-00646.html",
        "https://support.apple.com/kb/HT213189",
        "https://msrc-blog.microsoft.com/2021/12/11/microsofts-response-to-cve-2021-44228-apache-log4j2/",
        "https://logging.apache.org/log4j/2.x/security.html",

```

```
"https://www.debian.org/security/2021/dsa-5020",
"https://cert-portal.siemens.com/productcert/pdf/ssa-479842.pdf",
"https://www.oracle.com/security-alerts/alert-cve-2021-44228.html",
"https://www.oracle.com/security-alerts/cpujan2022.html",
"https://cert-portal.siemens.com/productcert/pdf/ssa-714170.pdf",
"https://lists.fedoraproject.org/archives/list/package-
announce@lists.fedoraproject.org/message/M5CSVUNV4HWZZXG0KNSK6L7RPM7B0KIB/",
"https://cert-portal.siemens.com/productcert/pdf/ssa-397453.pdf",
"https://cert-portal.siemens.com/productcert/pdf/ssa-661247.pdf",
"https://lists.fedoraproject.org/archives/list/package-
announce@lists.fedoraproject.org/message/VU57UJDCFIASI035GC55JMKSRXJMCDFM/",
"https://www.oracle.com/security-alerts/cpuapr2022.html",
"https://twitter.com/kurtseifried/status/1469345530182455296",
"https://tools.cisco.com/security/center/content/CiscoSecurityAdvisory/cisco-
sa-apache-log4j-qRuKNEbd",
"https://lists.debian.org/debian-lts-announce/2021/12/msg00007.html",
"https://www.kb.cert.org/vuls/id/930724"
],
"created": "2021-12-10T10:15:00Z",
"updated": "2023-04-03T20:15:00Z",
"properties": {
  "cisa_kev_date_added": "2021-12-10T00:00:00Z",
  "cisa_kev_date_due": "2021-12-24T00:00:00Z",
  "cwes": [
    400,
    20,
    502
  ],
},
"cvss": [
  {
    "source": "NVD",
    "severity": "critical",
    "cvss3_base_score": 10.0,
    "cvss3_base_vector": "AV:N/AC:L/PR:N/UI:N/S:C/C:H/I:H/A:H",
    "cvss2_base_score": 9.3,
    "cvss2_base_vector": "AC:M/Au:N/C:C/I:C/A:C"
  },
  {
    "source": "SNYK",
    "severity": "critical",
    "cvss3_base_score": 10.0,
    "cvss3_base_vector": "AV:N/AC:L/PR:N/UI:N/S:C/C:H/I:H/A:H/E:H"
  },
],
{
```

```
        "source": "GITHUB",
        "severity": "critical",
        "cvss3_base_score": 10.0,
        "cvss3_base_vector": "AV:N/AC:L/PR:N/UI:N/S:C/C:H/I:H/A:H"
    }
],
"epss": 0.97565,
"exploit_available": true,
"exploit_last_seen_in_public": "2023-03-06T00:00:00Z"
},
"affects": [
    {
        "installed_version": "pkg:maven/org.apache.logging.log4j/log4j-
core@2.12.1",
        "fixed_version": "2.15.0",
        "path": "/home/dev/foo.jar"
    }
]
}
]
```

Using the Amazon Inspector Jenkins plugin

The Jenkins plugin leverages the [Amazon Inspector SBOM Generator](#) binary and Amazon Inspector Scan API to produce detailed reports at the end of your build, so you can investigate and remediate risk before deployment. With the Amazon Inspector Jenkins plugin, you can add Amazon Inspector vulnerability scans to your Jenkins pipeline. Amazon Inspector vulnerability scans can be configured to pass or fail pipeline executions based on the number and severity of vulnerabilities detected. You can view the latest version of the Jenkins plugin in the Jenkins marketplace at <https://plugins.jenkins.io/amazon-inspector-image-scanner/>. The following steps describe how to set up the Amazon Inspector Jenkins plugin.

Important

Before completing the following steps, you must upgrade Jenkins to version 2.387.3 or higher for the plugin to run.

Step 1. Set up an AWS account

Configure an AWS account with an IAM role that allows access to the Amazon Inspector Scan API. For instructions, see [Setting up an AWS account to use the Amazon Inspector CI/CD integration](#).

Step 2. Install the Amazon Inspector Jenkins Plugin

The following procedure describes how to install the Amazon Inspector Jenkins plugin from the Jenkins dashboard.

1. From the Jenkins dashboard, choose **Manage Jenkins**, and then choose **Manage Plugins**.
2. Choose **Available**.
3. From the **Available** tab, search for **Amazon Inspector Scans**, and then install the plugin.

(Optional) Step 3. Add docker credentials to Jenkins

Note

Only add docker credentials if the docker image is in a private repository. Otherwise, skip this step.

The following procedure describes how to add docker credentials to Jenkins from the Jenkins dashboard.

1. From the Jenkins dashboard, choose **Manage Jenkins**, **Credentials**, and then **System**.
2. Choose **Global credentials** and then **Add credentials**.
3. For **Kind**, select **Username with password**.
4. For **Scope**, select **Global (Jenkins, nodes, items, all child items, etc)**.
5. Enter your details, and then choose **OK**.

(Optional) Step 4. Add AWS credentials

Note

Only add AWS credentials if you want to authenticate based on an IAM user. Otherwise, skip this step.

The following procedure describes how to add AWS credentials from the Jenkins dashboard.

1. From the Jenkins dashboard, choose **Manage Jenkins, Credentials**, and then **System**.
2. Choose **Global credentials** and then **Add credentials**.
3. For **Kind**, select **AWS Credentials**.
4. Enter your details, including your **Access Key ID** and **Secret Access Key**, and then choose **OK**.

Step 5. Add CSS support in a Jenkins script

The following procedure describes how to add CSS support in a Jenkins script.

1. Restart Jenkins.
2. From the Dashboard, choose **Manage Jenkins, Nodes, Built-In Node**, and then **Script Console**.
3. In the text box, add the line
`System.setProperty("hudson.model.DirectoryBrowserSupport.CSP", "")`, and then choose **Run**.

Step 6. Add Amazon Inspector Scan to your build

You can add Amazon Inspector Scan to your build by adding a build step in your project or by using the Jenkins declarative pipeline.

Amazon Inspector Scan to your build by adding a build step in your project

1. On the configuration page, scroll down to **Build Steps**, and choose **Add build step**. Then select **Amazon Inspector Scan**.
2. Choose between two inspector-sbomgen installation methods: **Automatic** or **Manual**.

- a. (Option 1) Choose **Automatic** to download the latest version of `inspector-sbomgen`. If you choose this method, make sure to select the CPU architecture that matches the system that executes the plugin.
- b. (Option 2) Choose **Manual** if you want to set up the Amazon Inspector SBOM Generator binary for scanning. If you choose this method, make sure to provide the full path to a previously downloaded version of `inspector-sbomgen`.

For more information, see [Installing Amazon Inspector SBOM Generator \(Sbomgen\)](#) in [Amazon Inspector SBOM Generator](#).

3. Complete the following to finish configuring the Amazon Inspector Scan build step:
 - a. Input your **Image Id**. The image can be local, remote, or archived. Image names should follow the Docker naming convention. If analyzing an exported image, provide the path to the expected tar file. See the following example Image Id paths:
 - i. For local or remote containers: `NAME[:TAG|@DIGEST]`
 - ii. For a tar file: `/path/to/image.tar`
 - b. Select an **AWS Region** to send the scan request through.
 - c. (Optional) For **Docker credentials**, select your Docker username. Do this only if your container image is in a private repository.
 - d. (Optional) You can provide the following supported AWS authentication methods:
 - i. (Optional) For **IAM role**, provide a role ARN (`arn:aws:iam::AccountNumber:role/RoleName`).
 - ii. (Optional) For **AWS credentials**, select **Id** to authenticate based on an IAM user.
 - iii. (Optional) For **AWS profile name**, provide the name of a profile to authenticate using a profile name.
 - e. (Optional) Specify the **Vulnerability thresholds** per severity. If the number you specify is exceeded during a scan, the image build will fail. If the values are all `0`, the build will succeed, regardless if any vulnerabilities are found.
4. Choose **Save**.

Add Amazon Inspector Scan to your build using the Jenkins declarative pipeline

You can add Amazon Inspector Scan to your build using the Jenkins declarative pipeline automatically or manually.

To automatically download the SBOMGen declarative pipeline

- To add Amazon Inspector Scan to a build, use the following example syntax. Based on your preferred OS architecture of the Amazon Inspector SBOM Generator download, replace *SBOMGEN_SOURCE* with `linuxAmd64` or `linuxArm64`. Replace *IMAGE_PATH* with the path to your image (such as `alpine:latest`), *IAM_ROLE* with the ARN of the IAM role you configured in step 1, and *ID* with your Docker credential ID if you are using a private repository. You can optionally enable vulnerability thresholds and specify values for each severity.

```
pipeline {
  agent any
  stages {
    stage('amazon-inspector-image-scanner') {
      steps {
        script {
          step([
            $class:
            'com.amazon.inspector.jenkins.amazoninspectorbuildstep.AmazonInspectorBuilder',
            sbomgenSource: 'SBOMGEN_SOURCE', // this can be linuxAmd64 or linuxArm64
            archivePath: 'IMAGE_PATH',
            awsRegion: 'REGION',
            iamRole: 'IAM_ROLE',
            credentialId: 'Id', // provide empty string if image not in private
            repositories
            awsCredentialId: 'AWS ID',
            awsProfileName: 'Profile Name',
            isThresholdEnabled: false,
            countCritical: 0,
            countHigh: 0,
            countLow: 10,
            countMedium: 5,
          ])
        }
      }
    }
  }
}
```



```
}

```

To manually download the SBOMGen declarative pipeline

- To add Amazon Inspector Scan to a build, use the following example syntax. Replace *SBOMGEN_PATH* with the path to the Amazon Inspector SBOM Generator you installed in step 3, *IMAGE_PATH* with the path to your image (such as *alpine:latest*), *IAM_ROLE* with the ARN of the IAM role you configured in step 1, and *ID* with your Docker credential ID if you are using a private repository. You can optionally enable vulnerability thresholds and specify values for each severity.

Note

Place S bomgen in Jenkins directory, and provide the path to the Jenkins directory in plugin (such as */opt/folder/arm64/inspector-sbomgen*).

```
pipeline {
  agent any
  stages {
    stage('amazon-inspector-image-scanner') {
      steps {
        script {
          step([
            $class:
'com.amazon.inspector.jenkins.amazoninspectorbuildstep.AmazonInspectorBuilder',
            sbomgenPath: 'SBOMGEN_PATH',
            archivePath: 'IMAGE_PATH',
            awsRegion: 'REGION',
            iamRole: 'IAM ROLE',
            awsCredentialId: 'AWS ID',
            credentialId: 'Id', // provide empty string if image not in private
repositories
            awsProfileName: 'Profile Name',
            isThresholdEnabled: false,
            countCritical: 0,
            countHigh: 0,
            countLow: 10,
            countMedium: 5,
          ])
        }
      }
    }
  }
}
```

```

    }
  }
}
}
}

```

Step 7. View your Amazon Inspector vulnerability report

1. Complete a new build of your project.
2. After the build completes, select an output format from the results. If you select HTML, you have the option to download a JSON SBOM or CSV version of the report. The following shows an example of an HTML report:

Inspector Vulnerability Report
Updated at 11/8/2023, 3:52:55 PM

[Download SBOM](#) [Download CSV](#)

SBOM parsed successfully, 7 vulnerabilities found.

Information

Image name	Image SHA
file:///Users/naveehal/Downloads/alpine.tar	sha256:5977be310a9d079b4febfe923ccd67daf778253cdbaddf2488259b3b7c5ef70

Vulnerability by severity

Critical	High	Medium	Low
1	4	2	0

All vulnerabilities (7)

Vulnerability Id	Severity	Component
CVE-2022-37434	Critical	pkg:apk/alpine/zlib@1.2.12-r1?arch=x86_64&distro=3.14.7
CVE-2022-4450	High	pkg:apk/alpine/openssl@1.1.1q-r0?arch=x86_64&distro=3.14.7
CVE-2023-0215	High	pkg:apk/alpine/openssl@1.1.1q-r0?arch=x86_64&distro=3.14.7
CVE-2023-0286	High	pkg:apk/alpine/openssl@1.1.1q-r0?arch=x86_64&distro=3.14.7
CVE-2023-0464	High	pkg:apk/alpine/openssl@1.1.1q-r0?arch=x86_64&distro=3.14.7
CVE-2022-4304	Medium	pkg:apk/alpine/openssl@1.1.1q-r0?arch=x86_64&distro=3.14.7
CVE-2023-0465	Medium	pkg:apk/alpine/openssl@1.1.1q-r0?arch=x86_64&distro=3.14.7

Troubleshooting

The following are common errors you can encounter when using the Amazon Inspector Scan plugin for Jenkins.

Failed to load credentials or sts exception error

Error:

InstanceProfileCredentialsProvider(): Failed to load credentials or sts exception.

Resolution

Get `aws_access_key_id` and `aws_secret_access_key` for your AWS account. Set up `aws_access_key_id` and `aws_secret_access_key` in `~/.aws/credentials`.

Failed to load image from tarball, local, or remote sources

Error:

```
2024/10/16 02:25:17 [ImageDownloadFailed]: failed to load image from tarball, local, or remote sources.
```

Note

This error can occur if the Jenkins plugin cannot read the container image, the container image isn't found in the Docker engine, and the container image isn't found in the remote container registry.

Resolution:

Verify the following;

- The Jenkins plugin user has read permissions to the image you wish to scan.
- The image you wish to scan is present in Docker engine.
- Your remote image URL is correct.
- You are authenticated to the remote registry (if applicable).

Inspector-sbomgen path error

Error:

```
Exception:com.amazon.inspector.jenkins.amazoninspectorbuildstep.exception.Sbomge  
There was an issue running inspector-sbomgen, is /opt/inspector/inspector-  
sbomgen the correct path?
```

Resolution:

Complete the following procedure to resolve the issue.

1. Place correct OS architecture Inspector-sbomgen in Jenkins directory For more information, see [Amazon Inspector SBOM Generator](#).
2. Grant executable permissions to the binary using the following command: `chmod +x inspector-sbomgen`.
3. Provide correct Jenkins machine path in plugin, such as `/opt/folder/arm64/inspector-sbomgen`.
4. Save config, and execute Jenkins job.

Using the Amazon Inspector TeamCity plugin

The Amazon Inspector TeamCity plugin leverages the Amazon Inspector SBOM Generator binary and Amazon Inspector Scan API to produce detailed reports at the end of your build, so you can investigate and remediate risk before deployment. With the Amazon Inspector TeamCity plugin, you can add Amazon Inspector vulnerability scans to your TeamCity pipeline. Amazon Inspector vulnerability scans can be configured to pass or fail pipeline executions based on the number and severity of vulnerabilities detected. You can view the latest version of the Amazon Inspector TeamCity plugin in the TeamCity marketplace at <https://plugins.jetbrains.com/plugin/23236-amazon-inspector-scanner>. For information about how to integrate Amazon Inspector Scan into your CI/CD pipeline, see [Integrating Amazon Inspector scans into your CI/CD pipeline](#). For a list of operating systems and programming languages that Amazon Inspector supports, see [Supported operating systems and programming languages](#). The following steps describe how to set up the Amazon Inspector TeamCity plugin.

1. **Set up an AWS account.**
 - Configure an AWS account with an IAM role that allows access to the Amazon Inspector Scan API. For instructions, see [Setting up an AWS account to use the Amazon Inspector CI/CD integration](#).
2. **Install the Amazon Inspector TeamCity plugin.**
 - a. From your dashboard, go to **Administration > Plugins**.
 - b. Search for **Amazon Inspector Scans**.
 - c. Install the plugin.
3. **Install the Amazon Inspector SBOM Generator.**

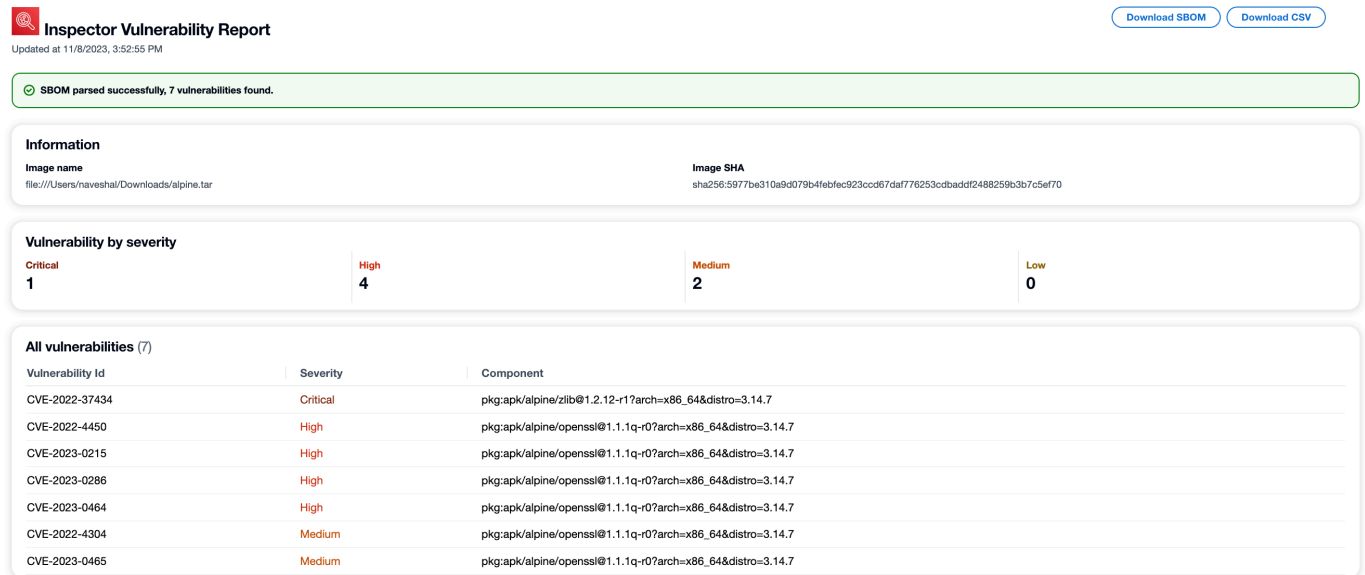
- Install the Amazon Inspector SBOM Generator binary in your Teamcity server directory. For instructions, see [Installing Sbmongen](#).
4. **Add an Amazon Inspector Scan build step to your project.**
- a. On the configuration page, scroll down to **Build Steps**, choose **Add build step**, and then select **Amazon Inspector Scan**.
 - b. Configure the Amazon Inspector Scan build step by filling in following details:
 - Add a **Step name**.
 - Choose between two Amazon Inspector SBOM Generator installation methods: **Automatic** or **Manual**.
 - **Automatic** downloads the most recent version of Amazon Inspector SBOM Generator based on your system and CPU architecture.
 - **Manual** requires that you provide a complete path to a previously downloaded version of Amazon Inspector SBOM Generator.

For more information, see [Installing Amazon Inspector SBOM Generator \(Sbmongen\) in Amazon Inspector SBOM Generator](#).

- Input your **Image Id**. Your image can be local, remote, or archived. Image names should follow the Docker naming convention. If analyzing an exported image, provide the path to the expected tar file. See the following example Image Id paths:
 - For local or remote containers: NAME [: TAG | @DIGEST]
 - For a tar file: /path/to/image.tar
 - For **IAM Role** enter the ARN for the role you configured in step 1.
 - Select an **AWS Region** to send the scan request through.
 - (Optional) For **Docker Authentication** enter your **Docker Username** and **Docker Password**. Do this only if your container image is in a private repository.
 - (Optional) For **AWS Authentication**, enter your AWS access key ID and AWS secret key. Do this only if you want to authenticate based on AWS credentials.
 - (Optional) Specify the **Vulnerability thresholds** per severity. If the number you specify is exceeded during a scan the image build will fail. If the values are all 0 the build will succeed regardless of the number of vulnerabilities found.
- c. Select **Save**.

5. View your Amazon Inspector vulnerability report.

- a. Complete a new build of your project.
- b. When the build completes select an output format from the results. When you select HTML you have the option to download a JSON SBOM or CSV version of the report. The following is an example of an HTML report:



Inspector Vulnerability Report
Updated at 11/8/2023, 3:52:55 PM

Download SBOM Download CSV

SBOM parsed successfully, 7 vulnerabilities found.

Information

Image name	Image SHA
file:///Users/naveshal/Downloads/alpine.tar	sha256:5977be310a9d079b4feb9c923ccd67daf776253c0baddf2488259b3b7c5ef70

Vulnerability by severity

Critical	High	Medium	Low
1	4	2	0

All vulnerabilities (7)

Vulnerability Id	Severity	Component
CVE-2022-37434	Critical	pkg:apk/alpine/zlib@1.2.12-r1?arch=x86_64&distro=3.14.7
CVE-2022-4450	High	pkg:apk/alpine/openssl@1.1.1q-r0?arch=x86_64&distro=3.14.7
CVE-2023-0215	High	pkg:apk/alpine/openssl@1.1.1q-r0?arch=x86_64&distro=3.14.7
CVE-2023-0286	High	pkg:apk/alpine/openssl@1.1.1q-r0?arch=x86_64&distro=3.14.7
CVE-2023-0464	High	pkg:apk/alpine/openssl@1.1.1q-r0?arch=x86_64&distro=3.14.7
CVE-2022-4304	Medium	pkg:apk/alpine/openssl@1.1.1q-r0?arch=x86_64&distro=3.14.7
CVE-2023-0465	Medium	pkg:apk/alpine/openssl@1.1.1q-r0?arch=x86_64&distro=3.14.7

Using Amazon Inspector with GitHub actions

You can use Amazon Inspector with [GitHub actions](#) to add Amazon Inspector vulnerability scans to your GitHub workflows. This leverages the [Amazon Inspector SBOM Generator](#) and [Amazon Inspector Scan API](#) to produce detailed reports at the end of your build, so you can investigate and remediate risk before deployment. Amazon Inspector vulnerability scans can be configured to pass or fail workflows based on the number and severity of vulnerabilities detected. You can view the latest version of the Amazon Inspector action on the [GitHub website](#). For information about how to integrate Amazon Inspector Scan into your CI/CD pipeline, see [Integrating Amazon Inspector scans into your CI/CD pipeline](#). For a list of operating systems and programming languages that Amazon Inspector supports, see [Supported operating systems and programming languages](#).

Using Amazon Inspector with GitLab components

You can use Amazon Inspector with [GitLab CI/CD components](#) to add Amazon Inspector vulnerability scans to your GitLab projects. This leverages the [Amazon Inspector SBOM Generator](#)

and [Amazon Inspector Scan API](#) to produce detailed reports at the end of your build, so you can investigate and remediate risk before deployment. Amazon Inspector vulnerability scans can be configured to pass or fail workflows based on the number and severity of vulnerabilities detected. You can view the latest version of the Amazon Inspector component on the [GitLab website](#). For information about how to integrate Amazon Inspector Scan into your CI/CD pipeline, see [Integrating Amazon Inspector scans into your CI/CD pipeline](#). For a list of operating systems and programming languages that Amazon Inspector supports, see [Supported operating systems and programming languages](#).

Using CodeCatalyst actions with Amazon Inspector

You can use Amazon Inspector with [Amazon CodeCatalyst](#) to add Amazon Inspector vulnerability scans to your CodeCatalyst workflows. This leverages the [Amazon Inspector SBOM Generator](#) and [Amazon Inspector Scan API](#) to produce detailed reports at the end of your build, so you can investigate and remediate risk before deployment. Amazon Inspector vulnerability scans can be configured to pass or fail workflows based on the number and severity of vulnerabilities detected. For information about how to integrate Amazon Inspector Scan into your CI/CD pipeline, see [Integrating Amazon Inspector scans into your CI/CD pipeline](#). For a list of operating systems and programming languages that Amazon Inspector supports, see [Supported operating systems and programming languages](#).

Using CycloneDX namespaces with Amazon Inspector

Amazon Inspector provides you with CycloneDX namespaces and property names that you can use with SBOMs. This section describes all of the custom key/value properties that might be added to components in CycloneDX SBOMs. For more information, see [CycloneDX property taxonomy](#) on the GitHub website.

amazon:inspector:sbom_scanner namespace taxonomy

The Amazon Inspector Scan API uses the `amazon:inspector:sbom_scanner` namespace and has the following properties:

Property	Description
<code>amazon:inspector:sbom_scanner:cisa_kev_date_added</code>	Indicates when the vulnerability was added to the CISA Known Exploited Vulnerabilities catalog.
<code>amazon:inspector:sbom_scanner:cisa_kev_date_due</code>	Indicates when the vulnerability fix is due according to the CISA Known Exploited Vulnerabilities catalog.
<code>amazon:inspector:sbom_scanner:critical_vulnerabilities</code>	Count of the total number of critical severity vulnerabilities found in the SBOM.
<code>amazon:inspector:sbom_scanner:exploit_available</code>	Indicates if an exploit is available for the given vulnerability.
<code>amazon:inspector:sbom_scanner:exploit_last_seen_in_public</code>	Indicates when an exploit was last seen in public for the given vulnerability.
<code>amazon:inspector:sbom_scanner:fixed_version: <i>component_bom_ref</i></code>	Provides the fixed version of the indicated component for the given vulnerability.
<code>amazon:inspector:sbom_scanner:high_vulnerabilities</code>	Count of the total number of high severity vulnerabilities found in the SBOM.
<code>amazon:inspector:sbom_scanner:info</code>	Provides scan context for a given component , for example: "Component scanned: no vulnerabilities found."
<code>amazon:inspector:sbom_scanner:is_malicious</code>	Indicates if OpenSSF identifies affected components as malicious.
<code>amazon:inspector:sbom_scanner:low_vulnerabilities</code>	Count of the total number of low severity vulnerabilities found in the SBOM.
<code>amazon:inspector:sbom_scanner:medium_vulnerabilities</code>	Count of the total number of medium severity vulnerabilities found in the SBOM.

Property	Description
<code>amazon:inspector:sbom_scanner:path</code>	The path to the file that yields the subject package information.
<code>amazon:inspector:sbom_scanner:priority</code>	The recommended priority for fixing a given vulnerability. The values in descending order are "IMMEDIATE", "URGENT", "MODERATE", and "STANDARD".
<code>amazon:inspector:sbom_scanner:priority_intelligence</code>	The quality of intelligence used to determine the priority for a given vulnerability. The values include "VERIFIED" or "UNVERIFIED".
<code>amazon:inspector:sbom_scanner:warning</code>	Provides context for a why a given component was not scanned, for example: "Component skipped: no purl provided."

amazon:inspector:sbom_generator namespace taxonomy

The Amazon Inspector SBOM Generator uses the `amazon:inspector:sbom_generator` namespace and has the following properties:

Property	Description
<code>amazon:inspector:sbom_generator:cpu_architecture</code>	The CPU architecture of the system being inventoried, such as <code>x86_64</code> .
<code>amazon:inspector:sbom_generator:dockerfile_finding: <i>inspector_finding_id</i></code>	Indicates that an Amazon Inspector finding in a component is related to Dockerfile checks.
<code>amazon:inspector:sbom_generator:expires_after</code>	The date after which the SSL certificate is invalid.
<code>amazon:inspector:sbom_generator:expires_before</code>	The date before the SSL certificate is valid.

Property	Description
amazon:inspector:sbom_generator:go_toolchain	Indicates the Go compiler or toolchain version used to produce a Go executable.
amazon:inspector:sbom_generator:image_id	The hash of the container image's config file, also known as the Image ID.
amazon:inspector:sbom_generator:is_duplicate_package	Indicates that the subject package was found by more than one file scanner.
amazon:inspector:sbom_generator:is_expired	A Boolean value that indicates if the SSL certificate has expired.
amazon:inspector:sbom_generator:kernel_name	The kernel name of the system being inventoried.
amazon:inspector:sbom_generator:kernel_version	The kernel version of the system being inventoried.
amazon:inspector:sbom_generator:layer_diff_id	The hash of the uncompressed container image layer.
amazon:inspector:sbom_generator:os_hostname	The hostname of the system being inventoried.
amazon:inspector:sbom_generator:source_file_scanner	The scanner that found the file that contains package information, for example: <code>/var/lib/dpkg/status</code> .
amazon:inspector:sbom_generator:source_package_collector	The collector that extracted the package name and version from a specific file.
amazon:inspector:sbom_generator:source_path	The path to the file that the subject package information was extracted from.

Assessing Amazon Inspector coverage of your AWS environment

You can assess Amazon Inspector coverage of your AWS environment from the **Account management** screen in the Amazon Inspector console, which shows details and statistics about the status of Amazon Inspector scans for your accounts and resources.

Note

If you're the delegated administrator for an organization, you can view details and statistics for all the accounts in the organization.

The following procedure describes how to assess coverage of your Amazon Inspector environment.

To assess Amazon Inspector coverage of your AWS environment

1. Sign in using your credentials, and then open the Amazon Inspector console at <https://console.aws.amazon.com/inspector/v2/home>.
2. From the navigation pane, choose **Account management**.
3. To review coverage, choose one of the following tabs:
 - Choose **Accounts** to review account-level coverage.
 - Choose **Instances** to review coverage for Amazon Elastic Compute Cloud (Amazon EC2) instances.
 - Choose **Container repositories** to review coverage of Amazon Elastic Container Registry (Amazon ECR) repositories.
 - Choose **Container images** to review coverage for Amazon ECR container images.
 - Choose **Lambda functions** to review coverage for Lambda functions.

The following topics describe the information each of these tabs provide.

Topics

- [Assessing account-level coverage](#)
- [Assessing coverage of Amazon EC2 instances](#)

- [Assessing coverage of Amazon ECR repositories](#)
- [Assessing coverage of Amazon ECR container images](#)
- [Assessing coverage of AWS Lambda functions](#)

Assessing account-level coverage

If your account is not part of an organization or is not the delegated Amazon Inspector administrator account for an organization, the **Accounts** tab provides information about your account and the status of resource scanning for your account. On this tab, you can activate or deactivate scanning for all or only specific types of resources for your account. For more information, see [Automated scan types in Amazon Inspector](#).

If your account is the delegated Amazon Inspector administrator account for an organization, the **Accounts** tab provides automatic activation settings for accounts in your organization, and it lists all the accounts in your organization. For each account, the list indicates whether Amazon Inspector is activated for the account and, if so, the resource scanning types that are activated for the account. As the delegated administrator, you can use this tab to change the automatic activation settings for your organization. You can also activate or deactivate specific types of resource scanning for individual member accounts. For more information, see [Activating Amazon Inspector scans for member accounts](#).

Assessing coverage of Amazon EC2 instances

The **Instances** tab shows Amazon EC2 instances in your AWS environment. The lists are organized into groups on the following tabs:

- **All** – Shows all the instances in your environment. The **Status** column indicates the current scanning status for an instance.
- **Scanning** – Shows all the instances that Amazon Inspector is actively monitoring and scanning in your environment.
- **Not scanning** – Shows all the instances that Amazon Inspector is not monitoring and scanning in your environment. The **Reason** column indicates why Amazon Inspector is not monitoring and scanning an instance.

An EC2 instance can appear on the **Not scanning** tab for any of several reasons. Amazon Inspector uses AWS Systems Manager (SSM) and the SSM Agent to automatically monitor

and scan your EC2 instances for vulnerabilities. If an instance does not have the SSM Agent running, does not have an AWS Identity and Access Management (IAM) role that supports Systems Manager, or is not running a supported operating system or architecture, Amazon Inspector cannot monitor and scan the instance. For more information, see [Scanning Amazon EC2 instances](#).

On each tab, the **Account** column specifies the AWS account that owns an instance.

EC2 instance tags – This column shows you the tags associated with the instance and can be used to determine if your instance has been excluded from scans by tags.

Operating system – This column shows you the operating system type, which can be WINDOWS, MAC, LINUX, or UNKNOWN.

Monitored using – This column shows whether Amazon Inspector is using the [agent-based](#) or [agentless](#) scan method on this instance.

Last scanned – This column shows you when Amazon Inspector last checked that resource for vulnerabilities. The frequency that Amazon Inspector performs scans depends on the scan method it's using to scan the instance.

To review additional details about an EC2 instance, choose the link in the **EC2 instance** column. Amazon Inspector then displays details about the instance and current findings for the instance. To review the details of a finding, choose the link in the **Title** column. For information about these details, see [Viewing details for your Amazon Inspector findings](#).

Scanning status values for Amazon EC2 instances

For an Amazon Elastic Compute Cloud (Amazon EC2) instance, the possible **Status** values are:

- **Actively monitoring** – Amazon Inspector is continuously monitoring and scanning the instance.
- **Agentless instance storage limit exceeded** – Amazon Inspector uses this status when the combined size of all volumes attached to an instance is greater than 1200 GB, or an instance has more than 8 volumes attached to it.
- **Agentless instance collection time limit exceeded** – Amazon Inspector times out while trying to run an agentless scan on an instance.
- **EC2 instance stopped** – Amazon Inspector paused scanning for the instance because the instance is in a stopped state. Any existing findings will persist until the instance is terminated. If the instance is restarted, Amazon Inspector will automatically resume scanning for the instance.

- **Internal error** – An internal error occurred when Amazon Inspector attempted to scan the instance. Amazon Inspector will automatically address the error and resume scanning as soon as possible.
- **No inventory** – Amazon Inspector couldn't find the software application inventory to scan for the instance. The Amazon Inspector associations for the instance might have been deleted or they might have failed to run.

To remediate this issue, use AWS Systems Manager to ensure that the `InspectorInventoryCollection-do-not-delete` association exists and its association status is successful. In addition, use AWS Systems Manager Fleet Manager to verify the software application inventory for the instance.

- **Pending disable** – Amazon Inspector has stopped scanning the instance. The instance is being disabled, pending completion of clean-up tasks.
- **Pending initial scan** – Amazon Inspector has queued the instance for an initial scan.
- **Resource terminated** – The instance was terminated. Amazon Inspector is currently cleaning up existing findings and coverage data for the instance.
- **Stale inventory** – Amazon Inspector wasn't able to collect an updated software application inventory that was captured within the past 7 days for the instance.

To remediate this issue, use AWS Systems Manager to ensure that the required Amazon Inspector associations exist and are running for the instance. In addition, use AWS Systems Manager Fleet Manager to verify the software application inventory for the instance.

- **Unmanaged EC2 instance** – Amazon Inspector isn't monitoring or scanning the instance. The instance isn't managed by AWS Systems Manager.

To remediate this issue, you can use the [AWSSupport-TroubleshootManagedInstance runbook](#) provided by AWS Systems Manager Automation. After you configure AWS Systems Manager to manage the instance, Amazon Inspector will automatically begin to continuously monitor and scan the instance.

- **Unsupported OS** – Amazon Inspector isn't monitoring or scanning the instance. The instance uses an operating system or architecture that Amazon Inspector doesn't support. For a list of operating systems that Amazon Inspector supports, see [Amazon EC2 instances status values](#).
- **Actively monitoring with partial errors** – This status means that EC2 scanning is active, but there are errors associated with [Amazon Inspector deep inspection for Linux-based Amazon EC2 instances](#). The possible deep inspections errors are:

- **Deep inspection package collection limit exceeded** – The instance has exceeded the 5000 package limit for Amazon Inspector deep inspection. To resume deep inspection for this instance, you can try to adjust the custom paths associated with the account.
- **Deep inspection daily ssm inventory limit exceeded** – The SSM agent couldn't send inventory to Amazon Inspector because the SSM quota for *Inventory data collected per instance per day* has already been reached for this instance. For more information, see [Amazon EC2 Systems Manager endpoints and quotas](#).
- **Deep inspection collection time limit exceeded** – Amazon Inspector failed to extract the package inventory because the package collection time exceeding the maximum threshold of 15 minutes.
- **Deep inspection has no inventory** – The [Amazon Inspector SSM plugin](#) hasn't yet been able to collect an inventory of packages for this instance. This is usually the result of a pending scan, however, if this status persists after 6 hours, use Amazon EC2 Systems Manager to ensure that the required Amazon Inspector associations exist and are running for the instance.

For details about configuring the scanning settings for an EC2 instance, see [Scanning Amazon EC2 instances](#).

Assessing coverage of Amazon ECR repositories

The **Repositories** tab shows Amazon ECR repositories in your AWS environment. The lists are organized into groups on the following tabs:

- **All** – Shows all the repositories in your environment. The **Status** column indicates the current scanning status for a repository.
- **Activated** – Shows all the repositories that Amazon Inspector is configured to monitor and scan in your environment. The **Status** column indicates the current scanning status for a repository.
- **Not activated** – Shows all the repositories that Amazon Inspector is not monitoring and scanning in your environment. The **Reason** column indicates why Amazon Inspector is not monitoring and scanning a repository.

On each tab, the **Account** column specifies the AWS account that owns a repository.

To review additional details about a repository, choose the repository's name. Amazon Inspector then displays a list of container images in the repository and details for each image. The details

include the image tag, image digest, and scanning status. They also include key finding statistics, such as the number of **Critical** findings for the image. To drill down and review supporting data for finding statistics, choose the image tag for the image.

Scanning status values for Amazon ECR repositories

For an Amazon Elastic Container Registry (Amazon ECR) repository, the possible **Status** values are:

- **Activated (Continuous)** – For a repository, Amazon Inspector is continuously monitoring images in this repository. The enhanced scanning setting for the repository is set to continuous scanning. Amazon Inspector initially scans new images when they are pushed and rescans images if a new CVE relevant to that image is published. Amazon Inspector will continue to be monitor images in this repository for the [Amazon ECR re-scan duration](#) you configure.
- **Activated (On push)** – Amazon Inspector automatically scans individual container images in the repository when a new image is pushed. Enhanced scanning is activated for the repository and set to scan on push.
- **Access denied** – Amazon Inspector isn't allowed to access the repository or any container images in the repository.

To remediate this issue, ensure that AWS Identity and Access Management (IAM) policies for the repository allow Amazon Inspector to access the repository.

- **Deactivated (Manual)** – Amazon Inspector isn't monitoring or scanning any container images in the repository. The Amazon ECR scanning setting for the repository is set to basic, manual scanning.

To start scanning images in the repository with Amazon Inspector, change the scanning setting for the repository to enhanced scanning, and then choose whether to scan images continuously or only when a new image is pushed.

- **Activated (On push)** – Amazon Inspector automatically scans individual container images in the repository when a new image is pushed. The enhanced scanning setting for the repository is set to scan on push.
- **Internal error** – An internal error occurred when Amazon Inspector attempted to scan the repository. Amazon Inspector will automatically address the error and resume scanning as soon as possible.

For details about configuring the scanning settings for repositories [Scanning Amazon ECR container images](#).

Assessing coverage of Amazon ECR container images

The **Images** tab shows Amazon ECR container images in your AWS environment. The lists are organized into groups on the following tabs:

- **All** – Shows all the container images in your environment. The **Status** column indicates the current scanning status for an image.
- **Scanning** – Shows all the container images that Amazon Inspector is configured to monitor and scan in your environment. The **Status** column indicates the current scanning status for an image.
- **Not scanning** – Shows all the container images that Amazon Inspector is not monitoring and scanning in your environment. The **Reason** column indicates why Amazon Inspector is not monitoring and scanning an image.

A container image can appear on the **Not activated** tab for any of several reasons. The image might be stored in a repository that Amazon Inspector scans are not activated for, or Amazon ECR filtering rules prevent that repository from being scanned. Or the image has not been pushed or pulled within the number of days you configured for the **ECR re-scan duration**. For more information, see [Configuring the Amazon ECR re-scan duration](#).

On each tab, the **Repository name** column specifies the name of the repository that stores a container image. The **Account** column specifies the AWS account that owns the repository. The **Last scanned** column shows you when Amazon Inspector last checked that resource for vulnerabilities. This can include checks when there is an update to finding metadata, when there is an update to the application inventory of the resource, or when a rescan is done in response to a new CVE. For more information, see [Scan behaviors for Amazon ECR scanning](#).

To review additional details about a container image, choose the link in the **ECR container image** column. Amazon Inspector then displays details about the image and current findings for the image. To review the details of a finding, choose the link in the **Title** column. For information about these details, see [Viewing details for your Amazon Inspector findings](#).

Scanning status values for Amazon ECR container images

For an Amazon Elastic Container Registry container image, the possible **Status** values are:

- **Actively monitoring (Continuous)** – Amazon Inspector is continuously monitoring and the image and new scans are performed on it whenever a new relevant CVE is published. The Amazon ECR

rescan duration for the image is refreshed whenever the image is pushed or pulled. Enhanced scanning is enabled for the repository that stores the image, and the enhanced scanning setting for the repository is set to continuous scanning.

- **Activated (On push)** – Amazon Inspector automatically scans the image each time a new image is pushed. Enhanced scanning is activated for the repository that stores the image, and the enhanced scanning setting for the repository is set to scan on push.
- **Internal error** – An internal error occurred when Amazon Inspector attempted to scan the container image. Amazon Inspector will automatically address the error and resume scanning as soon as possible.
- **Pending initial scan** – Amazon Inspector has queued the image for an initial scan.
- **Scan eligibility expired (Continuous)** – Amazon Inspector suspended scanning for the image. The image hasn't been updated within the duration that you specified for automated re-scans of images in the repository. You can push or pull the image to resume scanning.
- **Scan eligibility expired (On push)** – Amazon Inspector suspended scanning for the image. The image hasn't been updated within the duration that you specified for automated re-scans of images in the repository. You can push the image to resume scanning.
- **Scan frequency manual (Manual)** – Amazon Inspector doesn't scan the Amazon ECR container image. The Amazon ECR scanning setting for the repository that stores image is set to basic, manual scanning. To start scanning the image automatically with Amazon Inspector, change the repository setting to enhanced scanning, and then choose whether to scan images continuously or only when a new image is pushed.
- **Unsupported OS** – Amazon Inspector isn't monitoring or scanning the image. The image is based on an operating system that Amazon Inspector doesn't support, or it uses a media type that Amazon Inspector doesn't support.

For a list of operating systems that Amazon Inspector supports, see [Supported operating systems: Amazon ECR scanning with Amazon Inspector](#). For a list of media types that Amazon Inspector supports, see [Supported media types](#).

For details about configuring the scanning settings for repositories and images, see [Scanning Amazon ECR container images](#).

Assessing coverage of AWS Lambda functions

The **Lambda** tab shows Lambda functions in your AWS environment. This page two tables, one that shows function coverage details for Lambda standard scanning and another for Lambda code scanning. You can group functions based on the following tabs:

- **All** – Shows all the Lambda functions in your environment. The **Status** column indicates the current scanning status for a Lambda function.
- **Scanning** – Shows the Lambda functions that Amazon Inspector is configured to scan. The **Status** column indicates the current scanning status for each Lambda function.
- **Not scanning** – Shows the Lambda functions that Amazon Inspector is not configured to scan. The **Reason** column indicates why Amazon Inspector is not monitoring and scanning a function.

A Lambda function can appear on the **Not scanning** tab for several reasons. The Lambda function might belong to an account that hasn't been added to Amazon Inspector or filtering rules prevent this function from being scanned. For more information, see [Scanning Lambda functions](#).

On each tab, the **Function name** column specifies the name of the Lambda function. The **Account** column specifies the AWS account that owns the function. **Runtime** specifies the function's runtime. The **Status** column indicates the current scanning status for each Lambda function. **Resource tags** shows the tags that have been applied to the function. The **Last scanned** column shows you when Amazon Inspector last checked that resource for vulnerabilities. This can include checks when there is an update to finding metadata, when there is an update to the application inventory of the resource, or when a rescan is done in response to a new CVE. For more information, see [Scan behaviors for Lambda function scanning](#).

Scanning status values for AWS Lambda functions

For a Lambda function, the possible **Status** values are:

- **Actively monitoring** – Amazon Inspector is continuously monitoring and scanning Lambda functions. Continuous scanning includes an initial scan of new functions when they are pushed to the repository and automated re-scans of functions when they are updated or when new Common Vulnerabilities and Exposures (CVEs) are released.
- **Excluded by tag** – Amazon Inspector isn't scanning this function because it has been excluded from scans by tags.

- **Scan eligibility expired**– Amazon Inspector is not monitoring this function because it has been 90 days or more since it was last invoked or updated.
- **Internal error**–An internal error occurred when Amazon Inspector attempted to scan the function. Amazon Inspector will automatically address the error and resume scanning as soon as possible.
- **Pending initial scan**– Amazon Inspector has queued the function for an initial scan.
- **Unsupported**– The Lambda function has an unsupported runtime.

Managing multiple accounts in Amazon Inspector with AWS Organizations

You can use Amazon Inspector to manage multiple accounts associated with [AWS Organizations](#). To manage multiple Amazon Inspector accounts, the Organizations management account designates an account as the delegated administrator account for Amazon Inspector. The delegated administrator account manages Amazon Inspector for an organization and is granted special permissions to perform tasks on behalf of the organization. These tasks include activating or deactivating scans for member accounts, creating and managing suppression rules, and viewing aggregated finding for the entire organization.

Note

To enable Amazon Inspector programatically for multiple accounts in multiple AWS Regions, you can use a shell script that Amazon Inspector developed. For more information, see [inspector2-enablement-with-cli](#) on the GitHub website.

Topics

- [Understanding the delegated administrator account and member account in Amazon Inspector](#)
- [Designating a delegated administrator account for Amazon Inspector](#)

Understanding the delegated administrator account and member account in Amazon Inspector


When using Amazon Inspector in a multi-account environment, the delegated administrator account has access to specific metadata. The metadata includes standard scanning for Amazon EC2, Amazon ECR, and Lambda, and Lambda code scanning. It also includes security finding results for member accounts. This section provides information about which actions the delegated administrator account can make and member accounts can make.

Delegated administrator actions

Generally, when the delegated administrator applies settings to their account, those settings are applied to all of the other accounts in the organization. The delegated administrator can also view

and retrieve information for their own account and any associated member. An Amazon Inspector delegated administrator account can perform the following actions:

- View and manage the status of Amazon Inspector for associated accounts, including activating and deactivating Amazon Inspector.
- Activate or deactivate scanning types for all member accounts in the organization.
- View aggregated finding data across the organization and finding details for all member accounts within the organization.
- Create and manage suppression rules that apply to findings for all accounts in the organization.
- Activate Amazon ECR enhanced scanning for all members of the organization.
- View resource coverage for the entire organization.
- Define the duration for automated re-scans of ECR container images for all member accounts in the organization. The delegated administrator's scan duration setting overrides any setting that the member account previously set. All accounts in the organization share the Amazon ECR automated re-scan duration of the delegated administrators. You can't set different re-scan durations for individual accounts.
- Specify five custom paths for Amazon Inspector deep inspection for Amazon EC2 that will be used across all accounts in the organization. This is in addition to the five custom paths that a delegated administrator can set for their individual account. For more information about configuring deep inspection custom paths, see [Custom paths for Amazon Inspector deep inspection](#).
- Activate and deactivate Amazon Inspector deep inspection for member accounts.
- [Export SBOMs](#) for any member accounts in the organization.
- Set the Amazon EC2 scan mode for all member accounts in the organization. For more information, see [Managing scan mode](#).
- Create and manage CIS scan configurations for all accounts in the organization, except for any scan configurations created by member accounts.

 **Note**

If a member account leaves the organization, the delegated administrator will no longer be able to see scan configurations scheduled by that account.

- View CIS scan results for all accounts in the organization.

Member account actions

A member account can view and retrieve information about their account in Amazon Inspector, while settings for their account are managed by the delegated administrator. Member accounts within an organization can perform the following actions in Amazon Inspector:

- Activate Amazon Inspector for their own account.
- View resource coverage for their own account.
- View findings details for their own account.
- View the ECR container image automated re-scan duration setting for their own account.
- Specify five custom paths for Amazon Inspector deep inspection for EC2 that will be used for their individual account. These paths are scanned in addition to any custom paths that the delegated administrator has specified for the organization. For more information about configuring deep inspection paths, see [Custom paths for Amazon Inspector deep inspection](#).
- View the custom paths set by your delegated administrator for Amazon Inspector deep inspection.
- [Export SBOMs](#) for any resources associated with their account.
- View the scan mode for their account.
- Create and manage CIS scan configurations for their account.
- View the results of any CIS scans for resources in their account, including those scheduled by the delegated administrator.

Note

After activation, Amazon Inspector can be deactivated only by a delegated administrator account.

Designating a delegated administrator account for Amazon Inspector

The delegated administrator account is an account that you designate to manage all member accounts for an AWS service. Delegated administrator accounts in Amazon Inspector have access to specific metadata. For more information, see [Understanding the delegated administrator account](#)

[and member account Amazon Inspector](#). This section describes how to designate a delegated administrator for Amazon Inspector.

Important considerations for delegated administrators

Take note of the following factors that define how the delegated administrator operates in Amazon Inspector:

A delegated administrator can manage a maximum of 10,000 members.

Each Amazon Inspector delegated administrator has a quota of 10,000 member accounts. However, your organization could include more than 10,000 accounts. If you exceed 10,000 member accounts, you will receive a notification through the Amazon CloudWatch Personal Health Dashboard and an email to the delegated administrator account.

A delegated administrator is Regional.

Unlike AWS Organizations, Amazon Inspector is a Regional service. This means you must designate the a delegated administrator, add member accounts, and activate scan types in each AWS Region you want to use Amazon Inspector in.

An organization can have only one delegated administrator.

You can only have one delegated administrator for Amazon Inspector for an organization. If you have designated an account as a delegated administrator in one Region, that account must be your delegated administrator in all other Regions.

Changing a delegated administrator does not deactivate Amazon Inspector for member accounts.

If you remove the delegated administrator, Amazon Inspector won't be deactivated in those accounts, and scan settings won't be affected.

Your AWS Organization must have all features activated.

This is the default setting for AWS Organizations. If it's not activated, see [Activating all features in your organization](#).

Permissions required to designate a delegated administrator

You must have permission to activate Amazon Inspector and to designate an Amazon Inspector delegated administrator. Add the following statement to the end of an IAM policy to grant these permissions. For more information, see [Managing IAM policies](#).


```
{
  "Sid": "PermissionsForInspectorAdmin",
  "Effect": "Allow",
  "Action": [
    "inspector2:EnableDelegatedAdminAccount",
    "organizations:EnableAWSServiceAccess",
    "organizations:RegisterDelegatedAdministrator",
    "organizations:ListDelegatedAdministrators",
    "organizations:ListAWSServiceAccessForOrganization",
    "organizations:DescribeOrganizationalUnit",
    "organizations:DescribeAccount",
    "organizations:DescribeOrganization"
  ],
  "Resource": "*"
}
```

Designating a delegated administrator for your AWS organization

The following procedure shows you how to designate a delegated administrator for your AWS organization. When this designation is complete, Amazon Inspector is activated for both the Organizations management account and the chosen delegated administrator account.

Note

Only the Organizations management account can designate a delegated administrator.

Activating Amazon Inspector for the first time creates the service-linked role (SLR) `AWSServiceRoleForAmazonInspector` for the account. For more information about how Amazon Inspector uses service-linked roles, see [Using service-linked roles for Amazon Inspector](#). For information about service-linked roles in general, see [Using service-linked roles](#) in the *IAM User Guide*.

To designate a delegated administrator for Amazon Inspector

Console

Designate a delegated administrator in the console

1. Sign in to the AWS Management Console using the AWS Organizations management account.
2. Open the Amazon Inspector console at <https://console.aws.amazon.com/inspector/v2/home>.
3. Use the AWS Region selector to specify the Region where you want to designate a delegated administrator.
4. Choose **General settings**.
5. In the **Delegated administrator** tile, enter the account ID of the AWS account that you want to designate as the delegated administrator, and then choose **Delegate administration**.
6. (Optional) Repeat the previous steps for each AWS Region.

API

Designate a delegated administrator using the API

- Run the [EnableDelegatedAdminAccount](#) API operation using the credentials of the AWS account of the Organizations management account. You can also use the AWS Command Line Interface to do this by running the following CLI command:

```
aws inspector2 enable-delegated-admin-account --delegated-admin-account-id 111111111111.
```

Note

Make sure to specify the account ID of the account that you want to make an Amazon Inspector delegated administrator.

After you specify the delegated administrator, you must use the AWS Organizations management account only to change or remove the delegated administrator account.

Activating Amazon Inspector scans for member accounts

If you're the delegated administrator for an organization, you can activate Amazon EC2 and Amazon ECR scanning for member accounts in the organization. Once you activate scanning for a member account, Amazon Inspector is automatically activated for that account, and the account becomes associated with the delegated administrator account. For information about Amazon Inspector scanning types, see [Automated scan types in Amazon Inspector](#). This section describes how to activate scanning for member accounts.

Activate scanning for member accounts

You can activate scanning for member accounts in different ways. The following procedures describe how to activate scanning for all member accounts and specific member accounts as the delegated administrator, as well as how to activate scanning as a member account.

To automatically activate scanning for all member accounts

1. Sign in using the delegated administrator account credentials, and then open the Amazon Inspector console at <https://console.aws.amazon.com/inspector/v2/home>.
2. Use the region selector to choose the AWS Region where you want to activate scanning for all member accounts.
3. From the navigation pane, choose **Account management**. The **Accounts** tab displays all member accounts associated with the AWS Organizations management account.
4. Under **Organization**, select the box next to **Account number**. Then choose **Activate** to select which scanning options you want to apply to member accounts. You can select the following scanning types:
 - Amazon EC2 scanning
 - Amazon ECR scanning
 - Lambda standard scanning
 - Lambda code scanning
- After you select your preferred scanning types, choose **Save**.

Note

If you have multiple pages of accounts, you must repeat this step on each page. You can choose the gear icon to change the number of accounts displayed on each page.

5. Turn on the **Automatically activate Inspector for new member accounts** setting, and select which scanning options you want to apply to new member accounts added to your organization. You can select the following scanning types:
 - Amazon EC2 scanning
 - Amazon ECR scanning
 - Lambda standard scanning
 - Lambda code scanning
 - After you select your preferred scanning types, choose **Activate**.

Note


The **Automatically activate Inspector for new member accounts** setting activates Amazon Inspector for all future members of your organization. If the number of member accounts is more than 5,000, this setting is automatically turned off. If the total number of member accounts decreases to less than 5,000, the setting is automatically reactivated.

6. (Recommended) Repeat each of these steps in each AWS Region where you want to activate scanning for member accounts.

To activate scanning for specific member accounts

1. Sign in using the delegated administrator account credentials, and then open the Amazon Inspector console at <https://console.aws.amazon.com/inspector/v2/home>.
2. Use the region selector to choose the AWS Region where you want to activate scanning for all member accounts.

3. From the navigation pane, choose **Account management**. The **Accounts** tab displays all member accounts associated with the AWS Organizations management account.
4. Under **Organization**, select the box next to each member account number you want to activate scanning for. Then choose **Activate** to select which scanning options you want to apply to member accounts. You can select the following scanning types:
 - Amazon EC2 scanning
 - Amazon ECR scanning
 - Lambda standard scanning
 - Lambda code scanning
- After you select your preferred scanning types, choose **Save**.

 **Note**


If you have multiple pages of accounts, you must repeat this step on each page. You can choose the gear icon to change the number of accounts displayed on each page.

5. (Recommended) Repeat each of these steps in each AWS Region where you want to activate scanning for specific members.

To activate scanning as a member account

1. Sign in using your credentials, and then open the Amazon Inspector console at <https://console.aws.amazon.com/inspector/v2/home>.
2. Use the region selector to choose the AWS Region where you want to activate scanning for all member accounts.
3. From the navigation pane, choose **Account management**. The **Accounts** tab displays all member accounts associated with the AWS Organizations management account.
4. Under **Organization**, select the box next to your account number. Then choose **Activate** to select which scanning options you want to apply. You can select the following scanning types:
 - Amazon EC2 scanning
 - Amazon ECR scanning
 - Lambda standard scanning

- Lambda code scanning
 - After you select your preferred scanning types, choose **Save**.
5. (Recommended) Repeat these steps in each Region where you want to activate scanning for your member account.

 **Note**

If your AWS Organizations management account has a delegated administrator account for Amazon Inspector, you can activate your account as a member account to view scan details.

Disassociating member accounts in Amazon Inspector

As the delegated administrator, you might need to disassociate a member account from your account. When you disassociate a member account, Amazon Inspector is still activated in the account, and the account becomes a standalone account. You also don't have permission to manage Amazon Inspector for the account anymore. However, you can associate previously disassociated member accounts with your account at any time. This section describes how to disassociate member accounts as the delegated administrator.

Console

To disassociate member accounts using the console

1. Sign in using the delegated administrator account credentials, and then open the Amazon Inspector console at <https://console.aws.amazon.com/inspector/v2/home>
2. Use the region selector to choose the AWS Region where you want to disassociate member accounts.
3. From the navigation pane, choose **Account management**.
4. Under **Organization**, select the box next to each account number you want to disassociate.
5. Choose **Actions** menu, and then choose **Disassociate account**.

API

To disassociate member accounts using the API

Run the [DisassociateMember](#) API operation. In the request, provide the account IDs you're disassociating.

Removing the delegated administrator in Amazon Inspector

You might need to remove the Amazon Inspector delegated administrator account. You can do this from the AWS Organizations management account. When you remove the Amazon Inspector delegated administrator account, Amazon Inspector is still activated in the account and in all of its member accounts. The delegated administrator account and all of its member accounts become standalone accounts and retain their original scan settings. This section describes how to remove the delegated administrator account.

Remove the Amazon Inspector delegated administrator

The following procedures describe how to remove the Amazon Inspector delegated administrator and how to associate member accounts from the delegated administrator account.

For information about how to assign an Amazon Inspector delegated administrator, see [Designating a delegated administrator account for Amazon Inspector](#).

Note

After you assign an Amazon Inspector delegated administrator, the Amazon Inspector delegated administrator must associate member accounts manually.

To remove the delegated administrator

1. Sign in to the AWS Management Console using the AWS Organizations management account.
2. Open the Amazon Inspector console at <https://console.aws.amazon.com/inspector/v2/home>.
3. Use the region selector to choose the AWS Region where you want to remove the delegated administrator.
4. From the navigation pane, choose **General settings**.
5. Under **Delegated administrator**, choose **Remove**, and then confirm your action.

To associate members with a new delegated administrator

1. Sign in using the delegated administrator account credentials, and then open the Amazon Inspector console at <https://console.aws.amazon.com/inspector/v2/home>.
2. Use the region selector to choose the AWS Region where you want to associate members.
3. From the navigation pane, choose **Account management**.
4. Under **Organization**, select the box next to **Account number**.
5. Choose **Actions**, and then choose **Add member**.

Monitoring Usage and Cost in Amazon Inspector

You can use the Amazon Inspector console and API to project monthly Amazon Inspector costs for your environment. If you're the Amazon Inspector administrator for a multiple-account environment, you can view the total cost for your environment and cost metrics for all member accounts. This section describes how to access usage statistics and calculate usage costs.

Using the usage console

You can assess usage and projected cost for Amazon Inspector from the console.

To access usage statistics

1. Sign in using your credentials, and then open the Amazon Inspector console at <https://console.aws.amazon.com/inspector/v2/home>.
2. By using the AWS Region selector in the upper-right corner of the page, select the Region you want to monitor costs in.
3. In the navigation pane, choose **Usage**.

In the **By account** tab you will see the projected total cost based on the 30 day period listed under **Account usage**. In the table under the **Projected cost** column select a value to see a breakdown of usage by scan type for that account. In this detail pane you can also see which scan types have a free trial active for that account.

If you are the delegated administrator for an organization you will see a row in the table for each account within your organization. If an account in your organization is disassociated the console shows it's projected cost as a -.

In the **By scan type** tab you can see a break down of actual usage so far in the current 30 day period by scan type. This is the information used to calculate the projected costs in the **By account** tab.

If you are the delegated administrator for an organization you can see the usage for each account in your organization.

In this tab, you can expand any of the following panes for usage statistics:

Amazon EC2 scanning

The Amazon Inspector usage console tracks the following metrics for agent-based scanning and agentless scanning:

- **Instances (Avg)** — Amazon Inspector uses the coverage hours to calculate the average number of resources for EC2 instance scanning. The average is the total coverage hours divided by 720 hours (the number of hours in a 30 day period).
- **Coverage hours** — for Amazon EC2 scanning this is the sum total number of hours within the last 30 days that Amazon Amazon Inspector provided active coverage for each EC2 instance in an account. For EC2 instances, coverage hours are the hours from when Amazon Inspector discovered the instance until it's terminated or stopped, or excluded from scans by tags. (when you restart a stopped instance or remove an exclusion tag, Amazon Inspector resumes coverage and coverage hours for that instance will continue to accrue).

CIS instance Scans — The total number of CIS scans performed for instances in the account.

Amazon ECR scanning

Initial scans — The sum total of first time scans of images in the account within the last 30 days.

Rescans — The sum total of rescans for images in the account within the last 30 days. A rescan is any scan done on an ECR image that Amazon Inspector has previously scanned. If you have configured your ECR repository for continuous scanning, rescans occur automatically when Amazon Inspector adds a new Common Vulnerabilities and Exposures (CVE) to its database.

Lambda scanning

The Amazon Inspector usage console tracks the following metrics for Lambda standard scanning and Lambda code scanning:

- **Number of Lambda functions (Avg)** — Amazon Inspector uses the coverage hours to calculate the average number of functions for Lambda function scanning. Average is the total coverage hours divided by 720 hours (the number of hours in a 30 day period).
- **Coverage hours** — For Lambda function scanning this is the sum total number of hours within the last 30 days that Amazon Amazon Inspector provided active coverage for each Lambda function in an account. For AWS Lambda functions the coverage hours are calculated from when Amazon Inspector discovers a function until when it's deleted or excluded from scans. If an excluded function is included again, coverage hours for that function will continue to accrue.

Understanding how Amazon Inspector calculates usage costs

The costs provided by Amazon Inspector are estimates, not actual costs, so they may differ from those in your AWS Billing console.

Note the following about how Amazon Inspector calculates cost on the **Usage** page:

- The usage cost reflects the current region only. Prices per scan type vary by AWS Region, to review exact prices per region, see the [Pricing](#) for Amazon Inspector
- All usage projections are rounded to the nearest US dollar.
- Discounts aren't included in the projected costs.
- The projected cost represent the total cost for the 30 day usage period per scan type. If there has been less than 30 days of usage for an account, Amazon Inspector projects the cost after 30 days as if any currently covered resources will remain covered for the rest of the 30 day period.
- The cost per scan type is calculated based on the following:
 - EC2 scanning: cost reflects the average number of EC2 instances covered by Amazon Inspector in the last 30 days.
 - ECR container scanning: cost reflects the sum of the number of initial image scans + image rescans in the last 30 days.
 - Lambda standard scanning: cost reflects the average number of Lambda functions covered by Amazon Inspector in the last 30 days.
 - Lambda code scanning: cost reflects the average number of Lambda functions covered by Amazon Inspector in the last 30 days.

About the Amazon Inspector free trial

In Amazon Inspector, each [scan type](#) has a free trial. When you activate a scan type, you automatically enroll in a 15-day free trial for that scan type. Once the free trial starts, it automatically expires in 15 days, even if you deactivate the scan type.

Note

The free trial does not apply to [CIS scanning](#).

Security in Amazon Inspector

Cloud security at AWS is the highest priority. As an AWS customer, you benefit from data centers and network architectures that are 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 Amazon Inspector, 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 Amazon Inspector. The following topics show you how to configure Amazon Inspector to meet your security and compliance objectives. You also learn how to use other AWS services that help you to monitor and secure your Amazon Inspector resources.

Topics

- [Data protection in Amazon Inspector](#)
- [Identity and Access Management for Amazon Inspector](#)
- [Monitoring Amazon Inspector](#)
- [Compliance validation for Amazon Inspector](#)
- [Resilience in Amazon Inspector](#)
- [Infrastructure security in Amazon Inspector](#)
- [Incident response in Amazon Inspector](#)

Data protection in Amazon Inspector

The AWS [shared responsibility model](#) applies to data protection in Amazon Inspector. As described in this model, AWS is responsible for protecting the global infrastructure that runs all of the AWS Cloud. You are responsible for maintaining control over your content that is hosted on this infrastructure. You are also responsible for the security configuration and management tasks for the AWS services that you use. For more information about data privacy, see the [Data Privacy FAQ](#). For information about data protection in Europe, see the [AWS Shared Responsibility Model and GDPR](#) blog post on the *AWS Security Blog*.

For data protection purposes, we recommend that you protect AWS account credentials and set up individual users with AWS IAM Identity Center or AWS Identity and Access Management (IAM). That way, each user is given only the permissions necessary to fulfill their job duties. We also recommend that you secure your data in the following ways:

- Use multi-factor authentication (MFA) with each account.
- Use SSL/TLS to communicate with AWS resources. We require TLS 1.2 and recommend TLS 1.3.
- Set up API and user activity logging with AWS CloudTrail. For information about using CloudTrail trails to capture AWS activities, see [Working with CloudTrail trails](#) in the *AWS CloudTrail User Guide*.
- Use AWS encryption solutions, along with all default security controls within AWS services.
- Use advanced managed security services such as Amazon Macie, which assists in discovering and securing sensitive data that is stored in Amazon S3.
- If you require FIPS 140-3 validated cryptographic modules when accessing AWS through a command line interface or an API, use a FIPS endpoint. For more information about the available FIPS endpoints, see [Federal Information Processing Standard \(FIPS\) 140-3](#).

We strongly recommend that you never put confidential or sensitive information, such as your customers' email addresses, into tags or free-form text fields such as a **Name** field. This includes when you work with Amazon Inspector or other AWS services using the console, API, AWS CLI, or AWS SDKs. Any data that you enter into tags or free-form text fields used for names may be used for billing or diagnostic logs. If you provide a URL to an external server, we strongly recommend that you do not include credentials information in the URL to validate your request to that server.

Topics

- [Encryption at rest](#)

- [Encryption in transit](#)

Encryption at rest

By default, Amazon Inspector stores data at rest using AWS encryption solutions. Amazon Inspector encrypts data, such as the following:

- Resource inventory collected with AWS Systems Manager.
- Resource inventory parsed from Amazon Elastic Container Registry images
- Generated security findings using AWS owned encryption keys from AWS Key Management Service

You cannot manage, use, or view AWS owned keys. However, you don't need to take action or change programs to protect keys that encrypt your data. For more information, see [AWS owned keys](#).

If you disable Amazon Inspector, it permanently deletes all resources it stores or maintains for you, such as collected inventory and security findings.

Encryption at rest for code in your findings

For Amazon Inspector Lambda code scanning, Amazon Inspector partners with CodeGuru to scan your code for vulnerabilities. When a vulnerability is detected CodeGuru extracts a snippet of your code containing the vulnerability and stores that code until Amazon Inspector requests access. By default CodeGuru uses an AWS owned key to encrypt the extracted code, however, you can configure Amazon Inspector to use your own customer managed AWS KMS key for encryption.

The following work flow explains how Amazon Inspector uses the key you configure to encrypt your code:

1. You supply a AWS KMS key to Amazon Inspector using the Amazon Inspector [UpdateEncryptionKey](#) API.
2. Amazon Inspector forwards the information about your AWS KMS key to CodeGuru. CodeGuru stores the information for future use.
3. CodeGuru requests a [grant](#) from AWS KMS for the key you configured in Amazon Inspector.
4. CodeGuru creates an encrypted data key from your AWS KMS key and stores it. This data key is used to encrypt your code data stored by CodeGuru.

5. Whenever Amazon Inspector requests data from code scans CodeGuru uses the grant to decrypt the encrypted data key, then uses that key to decrypt the data so it can be retrieved.

When you disable Lambda code scanning CodeGuru retires the grant and deletes the associated data key.

Permissions for code encryption with a customer managed key

To use encryption you need to have a policy that allows access to AWS KMS actions, as well as a statement that grants Amazon Inspector and CodeGuru permissions to use those actions through condition keys.

If you are setting, updating, or resetting the encryption key for your account you will need to use an Amazon Inspector administrator policy, such as [AWS managed policy: AmazonInspector2FullAccess](#). You will also need to grant the following permissions to read-only users who need to retrieve code snippets from findings or data about the key chosen for encryption.

For KMS, the policy must allow you to perform the following actions:

- `kms:CreateGrant`
- `kms:Decrypt`
- `kms:DescribeKey`
- `kms:GenerateDataKeyWithoutPlainText`
- `kms:Encrypt`
- `kms:RetireGrant`

Once you've verified that you have the correct AWS KMS permissions in your policy, you must attach a statement that allows Amazon Inspector and CodeGuru to use your key for encryption. Attach the following policy statement:

Note

Replace Region with the AWS Region you have Amazon Inspector Lambda code scanning enabled in.

```
{
    "Sid": "allow CodeGuru Security to request a grant for a AWS KMS key",
    "Effect": "Allow",
    "Action": "kms:CreateGrant",
    "Resource": "*",
    "Condition": {
        "ForAllValues:StringEquals": {
            "kms:GrantOperations": [
                "GenerateDataKey",
                "GenerateDataKeyWithoutPlaintext",
                "Encrypt",
                "Decrypt",
                "RetireGrant",
                "DescribeKey"
            ]
        },
        "StringEquals": {
            "kms:ViaService": [
                "codeguru-security.Region.amazonaws.com"
            ]
        }
    },
},
{
    "Sid": "allow Amazon Inspector and CodeGuru Security to use your AWS KMS key",
    "Effect": "Allow",
    "Action": [
        "kms:Encrypt",
        "kms:Decrypt",
        "kms:RetireGrant",
        "kms:DescribeKey",
        "kms:GenerateDataKeyWithoutPlaintext"
    ],
    "Resource": "*",
    "Condition": {
        "StringEquals": {
            "kms:ViaService": [
                "inspector2.Region.amazonaws.com",
                "codeguru-security.Region.amazonaws.com"
            ]
        }
    }
}
```



```
}
```

Note

When you add the statement, ensure that the syntax is valid. Policies use JSON format. This means that you need to add a comma before or after the statement, depending on where you add the statement to the policy. If you add the statement as the last statement, add a comma after the closing brace for the preceding statement. If you add it as the first statement or between two existing statements, add a comma after the closing brace for the statement.

Configuring encryption with a customer managed key

To configure encryption for your account using a customer managed key you must be an Amazon Inspector administrator with the permissions outlined in [Permissions for code encryption with a customer managed key](#). Additionally you will need a AWS KMS key in the same AWS Region as your findings, or a [multi-region key](#). You can use an existing symmetric key in your account or create a symmetric customer managed key by using the AWS Management Console, or the AWS KMS APIs. For more information see [Creating symmetric encryption AWS KMS keys](#) in the AWS KMS user guide.

Using the Amazon Inspector API to configure encryption

To set a key for encryption the [UpdateEncryptionKey](#) operation of the Amazon Inspector API while signed in as an Amazon Inspector administrator. In the API request, use the `kmsKeyId` field to specify the ARN of the AWS KMS key you want to use. For `scanType` enter `CODE` and for `resourceType` enter `AWS_LAMBDA_FUNCTION`.

You can use [UpdateEncryptionKey](#) API to check view which AWS KMS key Amazon Inspector is using for encryption.

Note

If you attempt to use `GetEncryptionKey` when you haven't set a customer managed key the operation returns a `ResourceNotFoundException` error which means that an AWS owned key is being used for encryption.

If you delete or the key or change it's policy to deny access to Amazon Inspector or CodeGuru you will be unable to access your code vulnerability findings and Lambda code scanning will fail for your account.

You can use `ResetEncryptionKey` to resume using an AWS owned key to encrypt code extracted as part of your Amazon Inspector findings.

Encryption in transit

AWS encrypts all data in transit between AWS internal systems and other AWS services. AWS Systems Manager gathers telemetry data from customer-owned EC2 instances it sends to AWS over a Transport Layer Security (TLS)-protected channel for assessment. Amazon ECR and AWS Lambda function scan findings that are sent to Security Hub are encrypted using a TLS-protected channel. For more information, see [Data Protection in Systems Manager](#) to understand how SSM encrypts data in transit.

Identity and Access Management for Amazon Inspector

AWS Identity and Access Management (IAM) is an AWS service that helps an administrator securely control access to AWS resources. IAM administrators control who can be *authenticated* (signed in) and *authorized* (have permissions) to use Amazon Inspector resources. IAM is an AWS service that you can use with no additional charge.

Topics

- [Audience](#)
- [Authenticating with identities](#)
- [Managing access using policies](#)
- [How Amazon Inspector works with IAM](#)
- [Identity-based policy examples for Amazon Inspector](#)
- [AWS managed policies for Amazon Inspector](#)
- [Using service-linked roles for Amazon Inspector](#)
- [Troubleshooting Amazon Inspector identity and access](#)

Audience

How you use AWS Identity and Access Management (IAM) differs, depending on the work that you do in Amazon Inspector.

Service user – If you use the Amazon Inspector service to do your job, then your administrator provides you with the credentials and permissions that you need. As you use more Amazon Inspector features to do your work, you might need additional permissions. Understanding how access is managed can help you request the right permissions from your administrator. If you cannot access a feature in Amazon Inspector, see [Troubleshooting Amazon Inspector identity and access](#).

Service administrator – If you're in charge of Amazon Inspector resources at your company, you probably have full access to Amazon Inspector. It's your job to determine which Amazon Inspector features and resources your service users should access. You must then submit requests to your IAM administrator to change the permissions of your service users. Review the information on this page to understand the basic concepts of IAM. To learn more about how your company can use IAM with Amazon Inspector, see [How Amazon Inspector works with IAM](#).

IAM administrator – If you're an IAM administrator, you might want to learn details about how you can write policies to manage access to Amazon Inspector. To view example Amazon Inspector identity-based policies that you can use in IAM, see [Identity-based policy examples for Amazon Inspector](#).

Authenticating with identities

Authentication is how you sign in to AWS using your identity credentials. You must be *authenticated* (signed in to AWS) as the AWS account root user, as an IAM user, or by assuming an IAM role.

You can sign in to AWS as a federated identity by using credentials provided through an identity source. AWS IAM Identity Center (IAM Identity Center) users, your company's single sign-on authentication, and your Google or Facebook credentials are examples of federated identities. When you sign in as a federated identity, your administrator previously set up identity federation using IAM roles. When you access AWS by using federation, you are indirectly assuming a role.

Depending on the type of user you are, you can sign in to the AWS Management Console or the AWS access portal. For more information about signing in to AWS, see [How to sign in to your AWS account](#) in the *AWS Sign-In User Guide*.

If you access AWS programmatically, AWS provides a software development kit (SDK) and a command line interface (CLI) to cryptographically sign your requests by using your credentials. If you don't use AWS tools, you must sign requests yourself. For more information about using the recommended method to sign requests yourself, see [AWS Signature Version 4 for API requests](#) in the *IAM User Guide*.

Regardless of the authentication method that you use, you might be required to provide additional security information. For example, AWS recommends that you use multi-factor authentication (MFA) to increase the security of your account. To learn more, see [Multi-factor authentication](#) in the *AWS IAM Identity Center User Guide* and [AWS Multi-factor authentication in IAM](#) in the *IAM User Guide*.

AWS account root user

When you create an AWS account, you begin with one sign-in identity that has complete access to all AWS services and resources in the account. This identity is called the AWS account *root user* and is accessed by signing in with the email address and password that you used to create the account. We strongly recommend that you don't use the root user for your everyday tasks. Safeguard your root user credentials and use them to perform the tasks that only the root user can perform. For the complete list of tasks that require you to sign in as the root user, see [Tasks that require root user credentials](#) in the *IAM User Guide*.

Federated identity

As a best practice, require human users, including users that require administrator access, to use federation with an identity provider to access AWS services by using temporary credentials.

A *federated identity* is a user from your enterprise user directory, a web identity provider, the AWS Directory Service, the Identity Center directory, or any user that accesses AWS services by using credentials provided through an identity source. When federated identities access AWS accounts, they assume roles, and the roles provide temporary credentials.

For centralized access management, we recommend that you use AWS IAM Identity Center. You can create users and groups in IAM Identity Center, or you can connect and synchronize to a set of users and groups in your own identity source for use across all your AWS accounts and applications. For information about IAM Identity Center, see [What is IAM Identity Center?](#) in the *AWS IAM Identity Center User Guide*.

IAM users and groups

An [IAM user](#) is an identity within your AWS account that has specific permissions for a single person or application. Where possible, we recommend relying on temporary credentials instead of creating IAM users who have long-term credentials such as passwords and access keys. However, if you have specific use cases that require long-term credentials with IAM users, we recommend that you rotate access keys. For more information, see [Rotate access keys regularly for use cases that require long-term credentials](#) in the *IAM User Guide*.

An [IAM group](#) is an identity that specifies a collection of IAM users. You can't sign in as a group. You can use groups to specify permissions for multiple users at a time. Groups make permissions easier to manage for large sets of users. For example, you could have a group named *IAMAdmins* and give that group permissions to administer IAM resources.

Users are different from roles. A user is uniquely associated with one person or application, but a role is intended to be assumable by anyone who needs it. Users have permanent long-term credentials, but roles provide temporary credentials. To learn more, see [Use cases for IAM users](#) in the *IAM User Guide*.

IAM roles

An [IAM role](#) is an identity within your AWS account that has specific permissions. It is similar to an IAM user, but is not associated with a specific person. To temporarily assume an IAM role in the AWS Management Console, you can [switch from a user to an IAM role \(console\)](#). You can assume a role by calling an AWS CLI or AWS API operation or by using a custom URL. For more information about methods for using roles, see [Methods to assume a role](#) in the *IAM User Guide*.

IAM roles with temporary credentials are useful in the following situations:

- **Federated user access** – To assign permissions to a federated identity, you create a role and define permissions for the role. When a federated identity authenticates, the identity is associated with the role and is granted the permissions that are defined by the role. For information about roles for federation, see [Create a role for a third-party identity provider \(federation\)](#) in the *IAM User Guide*. If you use IAM Identity Center, you configure a permission set. To control what your identities can access after they authenticate, IAM Identity Center correlates the permission set to a role in IAM. For information about permission sets, see [Permission sets](#) in the *AWS IAM Identity Center User Guide*.
- **Temporary IAM user permissions** – An IAM user or role can assume an IAM role to temporarily take on different permissions for a specific task.

- **Cross-account access** – You can use an IAM role to allow someone (a trusted principal) in a different account to access resources in your account. Roles are the primary way to grant cross-account access. However, with some AWS services, you can attach a policy directly to a resource (instead of using a role as a proxy). To learn the difference between roles and resource-based policies for cross-account access, see [Cross account resource access in IAM](#) in the *IAM User Guide*.
- **Cross-service access** – Some AWS services use features in other AWS services. For example, when you make a call in a service, it's common for that service to run applications in Amazon EC2 or store objects in Amazon S3. A service might do this using the calling principal's permissions, using a service role, or using a service-linked role.
 - **Forward access sessions (FAS)** – When you use an IAM user or role to perform actions in AWS, you are considered a principal. When you use some services, you might perform an action that then initiates another action in a different service. FAS uses the permissions of the principal calling an AWS service, combined with the requesting AWS service to make requests to downstream services. FAS requests are only made when a service receives a request that requires interactions with other AWS services or resources to complete. In this case, you must have permissions to perform both actions. For policy details when making FAS requests, see [Forward access sessions](#).
 - **Service role** – A service role is an [IAM role](#) that a service assumes to perform actions on your behalf. An IAM administrator can create, modify, and delete a service role from within IAM. For more information, see [Create a role to delegate permissions to an AWS service](#) in the *IAM User Guide*.
 - **Service-linked role** – A service-linked role is a type of service role that is linked to an AWS service. The service can assume the role to perform an action on your behalf. Service-linked roles appear in your AWS account and are owned by the service. An IAM administrator can view, but not edit the permissions for service-linked roles.
- **Applications running on Amazon EC2** – You can use an IAM role to manage temporary credentials for applications that are running on an EC2 instance and making AWS CLI or AWS API requests. This is preferable to storing access keys within the EC2 instance. To assign an AWS role to an EC2 instance and make it available to all of its applications, you create an instance profile that is attached to the instance. An instance profile contains the role and enables programs that are running on the EC2 instance to get temporary credentials. For more information, see [Use an IAM role to grant permissions to applications running on Amazon EC2 instances](#) in the *IAM User Guide*.

Managing access using policies

You control access in AWS by creating policies and attaching them to AWS identities or resources. A policy is an object in AWS that, when associated with an identity or resource, defines their permissions. AWS evaluates these policies when a principal (user, root user, or role session) makes a request. Permissions in the policies determine whether the request is allowed or denied. Most policies are stored in AWS as JSON documents. For more information about the structure and contents of JSON policy documents, see [Overview of JSON policies](#) in the *IAM User Guide*.

Administrators can use AWS JSON policies to specify who has access to what. That is, which **principal** can perform **actions** on what **resources**, and under what **conditions**.

By default, users and roles have no permissions. To grant users permission to perform actions on the resources that they need, an IAM administrator can create IAM policies. The administrator can then add the IAM policies to roles, and users can assume the roles.

IAM policies define permissions for an action regardless of the method that you use to perform the operation. For example, suppose that you have a policy that allows the `iam:GetRole` action. A user with that policy can get role information from the AWS Management Console, the AWS CLI, or the AWS API.

Identity-based policies

Identity-based policies are JSON permissions policy documents that you can attach to an identity, such as an IAM user, group of users, or role. These policies control what actions users and roles can perform, on which resources, and under what conditions. To learn how to create an identity-based policy, see [Define custom IAM permissions with customer managed policies](#) in the *IAM User Guide*.

Identity-based policies can be further categorized as *inline policies* or *managed policies*. Inline policies are embedded directly into a single user, group, or role. Managed policies are standalone policies that you can attach to multiple users, groups, and roles in your AWS account. Managed policies include AWS managed policies and customer managed policies. To learn how to choose between a managed policy or an inline policy, see [Choose between managed policies and inline policies](#) in the *IAM User Guide*.

Resource-based policies

Resource-based policies are JSON policy documents that you attach to a resource. Examples of resource-based policies are IAM *role trust policies* and Amazon S3 *bucket policies*. In services that

support resource-based policies, service administrators can use them to control access to a specific resource. For the resource where the policy is attached, the policy defines what actions a specified principal can perform on that resource and under what conditions. You must [specify a principal](#) in a resource-based policy. Principals can include accounts, users, roles, federated users, or AWS services.

Resource-based policies are inline policies that are located in that service. You can't use AWS managed policies from IAM in a resource-based policy.

Access control lists (ACLs)

Access control lists (ACLs) control which principals (account members, users, or roles) have permissions to access a resource. ACLs are similar to resource-based policies, although they do not use the JSON policy document format.

Amazon S3, AWS WAF, and Amazon VPC are examples of services that support ACLs. To learn more about ACLs, see [Access control list \(ACL\) overview](#) in the *Amazon Simple Storage Service Developer Guide*.

Other policy types

AWS supports additional, less-common policy types. These policy types can set the maximum permissions granted to you by the more common policy types.

- **Permissions boundaries** – A permissions boundary is an advanced feature in which you set the maximum permissions that an identity-based policy can grant to an IAM entity (IAM user or role). You can set a permissions boundary for an entity. The resulting permissions are the intersection of an entity's identity-based policies and its permissions boundaries. Resource-based policies that specify the user or role in the `Principal` field are not limited by the permissions boundary. An explicit deny in any of these policies overrides the allow. For more information about permissions boundaries, see [Permissions boundaries for IAM entities](#) in the *IAM User Guide*.
- **Service control policies (SCPs)** – SCPs are JSON policies that specify the maximum permissions for an organization or organizational unit (OU) in AWS Organizations. AWS Organizations is a service for grouping and centrally managing multiple AWS accounts that your business owns. If you enable all features in an organization, then you can apply service control policies (SCPs) to any or all of your accounts. The SCP limits permissions for entities in member accounts, including each AWS account root user. For more information about Organizations and SCPs, see [Service control policies](#) in the *AWS Organizations User Guide*.

- **Session policies** – Session policies are advanced policies that you pass as a parameter when you programmatically create a temporary session for a role or federated user. The resulting session's permissions are the intersection of the user or role's identity-based policies and the session policies. Permissions can also come from a resource-based policy. An explicit deny in any of these policies overrides the allow. For more information, see [Session policies](#) in the *IAM User Guide*.

Multiple policy types

When multiple types of policies apply to a request, the resulting permissions are more complicated to understand. To learn how AWS determines whether to allow a request when multiple policy types are involved, see [Policy evaluation logic](#) in the *IAM User Guide*.

How Amazon Inspector works with IAM

Before you use IAM to manage access to Amazon Inspector, learn what IAM features are available to use with Amazon Inspector.

IAM features you can use with Amazon Inspector

IAM feature	Amazon Inspector support
Identity-based policies	Yes
Resource-based policies	No
Policy actions	Yes
Policy resources	Yes
Policy condition keys (service-specific)	Yes
ACLs	No
ABAC (tags in policies)	Partial
Temporary credentials	Yes
Principal permissions	Yes
Service roles	No

IAM feature	Amazon Inspector support
Service-linked roles	Yes

To get a high-level view of how Amazon Inspector and other AWS services work with most IAM features, see [AWS services that work with IAM](#) in the *IAM User Guide*.

Identity-based policies for Amazon Inspector

Supports identity-based policies: Yes

Identity-based policies are JSON permissions policy documents that you can attach to an identity, such as an IAM user, group of users, or role. These policies control what actions users and roles can perform, on which resources, and under what conditions. To learn how to create an identity-based policy, see [Define custom IAM permissions with customer managed policies](#) in the *IAM User Guide*.

With IAM identity-based policies, you can specify allowed or denied actions and resources as well as the conditions under which actions are allowed or denied. You can't specify the principal in an identity-based policy because it applies to the user or role to which it is attached. To learn about all of the elements that you can use in a JSON policy, see [IAM JSON policy elements reference](#) in the *IAM User Guide*.

Identity-based policy examples for Amazon Inspector

To view examples of Amazon Inspector identity-based policies, see [Identity-based policy examples for Amazon Inspector](#).

Resource-based policies within Amazon Inspector

Supports resource-based policies: No

Resource-based policies are JSON policy documents that you attach to a resource. Examples of resource-based policies are IAM *role trust policies* and Amazon S3 *bucket policies*. In services that support resource-based policies, service administrators can use them to control access to a specific resource. For the resource where the policy is attached, the policy defines what actions a specified principal can perform on that resource and under what conditions. You must [specify a principal](#) in a resource-based policy. Principals can include accounts, users, roles, federated users, or AWS services.

To enable cross-account access, you can specify an entire account or IAM entities in another account as the principal in a resource-based policy. Adding a cross-account principal to a resource-based policy is only half of establishing the trust relationship. When the principal and the resource are in different AWS accounts, an IAM administrator in the trusted account must also grant the principal entity (user or role) permission to access the resource. They grant permission by attaching an identity-based policy to the entity. However, if a resource-based policy grants access to a principal in the same account, no additional identity-based policy is required. For more information, see [Cross account resource access in IAM](#) in the *IAM User Guide*.

Policy actions for Amazon Inspector

Supports policy actions: Yes

Administrators can use AWS JSON policies to specify who has access to what. That is, which **principal** can perform **actions** on what **resources**, and under what **conditions**.

The `Action` element of a JSON policy describes the actions that you can use to allow or deny access in a policy. Policy actions usually have the same name as the associated AWS API operation. There are some exceptions, such as *permission-only actions* that don't have a matching API operation. There are also some operations that require multiple actions in a policy. These additional actions are called *dependent actions*.

Include actions in a policy to grant permissions to perform the associated operation.

To see a list of Amazon Inspector actions, see [Actions defined by Amazon Inspector](#) in the *Service Authorization Reference*.

Policy actions in Amazon Inspector use the following prefix before the action:

```
inspector2
```

To specify multiple actions in a single statement, separate them with commas.

```
"Action": [  
  "inspector2:action1",  
  "inspector2:action2"  
]
```

To view examples of Amazon Inspector identity-based policies, see [Identity-based policy examples for Amazon Inspector](#).

Policy resources for Amazon Inspector

Supports policy resources: Yes

Administrators can use AWS JSON policies to specify who has access to what. That is, which **principal** can perform **actions** on what **resources**, and under what **conditions**.

The Resource JSON policy element specifies the object or objects to which the action applies. Statements must include either a Resource or a NotResource element. As a best practice, specify a resource using its [Amazon Resource Name \(ARN\)](#). You can do this for actions that support a specific resource type, known as *resource-level permissions*.

For actions that don't support resource-level permissions, such as listing operations, use a wildcard (*) to indicate that the statement applies to all resources.

```
"Resource": "*" 
```

To see a list of Amazon Inspector resource types and their ARNs, see [Resources defined by Amazon Inspector](#) in the *Service Authorization Reference*. To learn with which actions you can specify the ARN of each resource, see [Actions defined by Amazon Inspector](#).

To view examples of Amazon Inspector identity-based policies, see [Identity-based policy examples for Amazon Inspector](#).

Policy condition keys for Amazon Inspector

Supports service-specific policy condition keys: Yes

Administrators can use AWS JSON policies to specify who has access to what. That is, which **principal** can perform **actions** on what **resources**, and under what **conditions**.

The Condition element (or Condition *block*) lets you specify conditions in which a statement is in effect. The Condition element is optional. You can create conditional expressions that use [condition operators](#), such as equals or less than, to match the condition in the policy with values in the request.

If you specify multiple `Condition` elements in a statement, or multiple keys in a single `Condition` element, AWS evaluates them using a logical AND operation. If you specify multiple values for a single condition key, AWS evaluates the condition using a logical OR operation. All of the conditions must be met before the statement's permissions are granted.

You can also use placeholder variables when you specify conditions. For example, you can grant an IAM user permission to access a resource only if it is tagged with their IAM user name. For more information, see [IAM policy elements: variables and tags](#) in the *IAM User Guide*.

AWS supports global condition keys and service-specific condition keys. To see all AWS global condition keys, see [AWS global condition context keys](#) in the *IAM User Guide*.

To see a list of Amazon Inspector condition keys, see [Condition keys for Amazon Inspector](#) in the *Service Authorization Reference*. To learn with which actions and resources you can use a condition key, see [Actions defined by Amazon Inspector](#).

To view examples of Amazon Inspector identity-based policies, see [Identity-based policy examples for Amazon Inspector](#).

ACLs in Amazon Inspector

Supports ACLs: No

Access control lists (ACLs) control which principals (account members, users, or roles) have permissions to access a resource. ACLs are similar to resource-based policies, although they do not use the JSON policy document format.

ABAC with Amazon Inspector

Supports ABAC (tags in policies): Partial

Attribute-based access control (ABAC) is an authorization strategy that defines permissions based on attributes. In AWS, these attributes are called *tags*. You can attach tags to IAM entities (users or roles) and to many AWS resources. Tagging entities and resources is the first step of ABAC. Then you design ABAC policies to allow operations when the principal's tag matches the tag on the resource that they are trying to access.

ABAC is helpful in environments that are growing rapidly and helps with situations where policy management becomes cumbersome.

To control access based on tags, you provide tag information in the [condition element](#) of a policy using the `aws:ResourceTag/key-name`, `aws:RequestTag/key-name`, or `aws:TagKeys` condition keys.

If a service supports all three condition keys for every resource type, then the value is **Yes** for the service. If a service supports all three condition keys for only some resource types, then the value is **Partial**.

For more information about ABAC, see [Define permissions with ABAC authorization](#) in the *IAM User Guide*. To view a tutorial with steps for setting up ABAC, see [Use attribute-based access control \(ABAC\)](#) in the *IAM User Guide*.

Using temporary credentials with Amazon Inspector

Supports temporary credentials: Yes

Some AWS services don't work when you sign in using temporary credentials. For additional information, including which AWS services work with temporary credentials, see [AWS services that work with IAM](#) in the *IAM User Guide*.

You are using temporary credentials if you sign in to the AWS Management Console using any method except a user name and password. For example, when you access AWS using your company's single sign-on (SSO) link, that process automatically creates temporary credentials. You also automatically create temporary credentials when you sign in to the console as a user and then switch roles. For more information about switching roles, see [Switch from a user to an IAM role \(console\)](#) in the *IAM User Guide*.

You can manually create temporary credentials using the AWS CLI or AWS API. You can then use those temporary credentials to access AWS. AWS recommends that you dynamically generate temporary credentials instead of using long-term access keys. For more information, see [Temporary security credentials in IAM](#).

Cross-service principal permissions for Amazon Inspector

Supports forward access sessions (FAS): Yes

When you use an IAM user or role to perform actions in AWS, you are considered a principal. When you use some services, you might perform an action that then initiates another action in a different service. FAS uses the permissions of the principal calling an AWS service, combined with the requesting AWS service to make requests to downstream services. FAS requests are only made when a service receives a request that requires interactions with other AWS services or resources to

complete. In this case, you must have permissions to perform both actions. For policy details when making FAS requests, see [Forward access sessions](#).

Service roles for Amazon Inspector

Supports service roles: No

A service role is an [IAM role](#) that a service assumes to perform actions on your behalf. An IAM administrator can create, modify, and delete a service role from within IAM. For more information, see [Create a role to delegate permissions to an AWS service](#) in the *IAM User Guide*.

Warning

Changing the permissions for a service role might break Amazon Inspector functionality. Edit service roles only when Amazon Inspector provides guidance to do so.

Service-linked roles for Amazon Inspector

Supports service-linked roles: Yes

A service-linked role is a type of service role that is linked to an AWS service. The service can assume the role to perform an action on your behalf. Service-linked roles appear in your AWS account and are owned by the service. An IAM administrator can view, but not edit the permissions for service-linked roles.

For details about creating or managing service-linked roles, see [AWS services that work with IAM](#). Find a service in the table that includes a Yes in the **Service-linked role** column. Choose the **Yes** link to view the service-linked role documentation for that service.

Identity-based policy examples for Amazon Inspector

By default, users and roles don't have permission to create or modify Amazon Inspector resources. They also can't perform tasks by using the AWS Management Console, AWS Command Line Interface (AWS CLI), or AWS API. To grant users permission to perform actions on the resources that they need, an IAM administrator can create IAM policies. The administrator can then add the IAM policies to roles, and users can assume the roles.

To learn how to create an IAM identity-based policy by using these example JSON policy documents, see [Create IAM policies \(console\)](#) in the *IAM User Guide*.

For details about actions and resource types defined by Amazon Inspector, including the format of the ARNs for each of the resource types, see [Actions, resources, and condition keys for Amazon Inspector](#) in the *Service Authorization Reference*.

Topics

- [Policy best practices](#)
- [Using the Amazon Inspector console](#)
- [Allow users to view their own permissions](#)
- [Allow read-only access to all Amazon Inspector resources](#)
- [Allow full access to all Amazon Inspector resources](#)

Policy best practices

Identity-based policies determine whether someone can create, access, or delete Amazon Inspector resources in your account. These actions can incur costs for your AWS account. When you create or edit identity-based policies, follow these guidelines and recommendations:

- **Get started with AWS managed policies and move toward least-privilege permissions** – To get started granting permissions to your users and workloads, use the *AWS managed policies* that grant permissions for many common use cases. They are available in your AWS account. We recommend that you reduce permissions further by defining AWS customer managed policies that are specific to your use cases. For more information, see [AWS managed policies](#) or [AWS managed policies for job functions](#) in the *IAM User Guide*.
- **Apply least-privilege permissions** – When you set permissions with IAM policies, grant only the permissions required to perform a task. You do this by defining the actions that can be taken on specific resources under specific conditions, also known as *least-privilege permissions*. For more information about using IAM to apply permissions, see [Policies and permissions in IAM](#) in the *IAM User Guide*.
- **Use conditions in IAM policies to further restrict access** – You can add a condition to your policies to limit access to actions and resources. For example, you can write a policy condition to specify that all requests must be sent using SSL. You can also use conditions to grant access to service actions if they are used through a specific AWS service, such as AWS CloudFormation. For more information, see [IAM JSON policy elements: Condition](#) in the *IAM User Guide*.
- **Use IAM Access Analyzer to validate your IAM policies to ensure secure and functional permissions** – IAM Access Analyzer validates new and existing policies so that the policies adhere to the IAM policy language (JSON) and IAM best practices. IAM Access Analyzer provides

more than 100 policy checks and actionable recommendations to help you author secure and functional policies. For more information, see [Validate policies with IAM Access Analyzer](#) in the *IAM User Guide*.

- **Require multi-factor authentication (MFA)** – If you have a scenario that requires IAM users or a root user in your AWS account, turn on MFA for additional security. To require MFA when API operations are called, add MFA conditions to your policies. For more information, see [Secure API access with MFA](#) in the *IAM User Guide*.

For more information about best practices in IAM, see [Security best practices in IAM](#) in the *IAM User Guide*.

Using the Amazon Inspector console

To access the Amazon Inspector console, you must have a minimum set of permissions. These permissions must allow you to list and view details about the Amazon Inspector resources in your AWS account. If you create an identity-based policy that is more restrictive than the minimum required permissions, the console won't function as intended for entities (users or roles) with that policy.

You don't need to allow minimum console permissions for users that are making calls only to the AWS CLI or the AWS API. Instead, allow access to only the actions that match the API operation that they're trying to perform.

To ensure that users and roles can still use the Amazon Inspector console, also attach the Amazon Inspector *ConsoleAccess* or *ReadOnly* AWS managed policy to the entities. For more information, see [Adding permissions to a user](#) in the *IAM User Guide*.

Allow users to view their own permissions

This example shows how you might create a policy that allows IAM users to view the inline and managed policies that are attached to their user identity. This policy includes permissions to complete this action on the console or programmatically using the AWS CLI or AWS API.

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

```

    "Action": [
      "iam:GetUserPolicy",
      "iam:ListGroupsForUser",
      "iam:ListAttachedUserPolicies",
      "iam:ListUserPolicies",
      "iam:GetUser"
    ],
    "Resource": ["arn:aws:iam::*:user/${aws:username}"]
  },
  {
    "Sid": "NavigateInConsole",
    "Effect": "Allow",
    "Action": [
      "iam:GetGroupPolicy",
      "iam:GetPolicyVersion",
      "iam:GetPolicy",
      "iam:ListAttachedGroupPolicies",
      "iam:ListGroupPolicies",
      "iam:ListPolicyVersions",
      "iam:ListPolicies",
      "iam:ListUsers"
    ],
    "Resource": "*"
  }
]
}

```

Allow read-only access to all Amazon Inspector resources

This example shows a policy that allows read-only access to all Amazon Inspector resources.

```

{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "inspector2:Describe*",
        "inspector2:Get*",
        "inspector2:BatchGet*",
        "inspector2:List*"
      ],
      "Resource": "*"
    }
  ]
}

```

```

    },
    {
      "Effect": "Allow",
      "Action": [
        "organizations:ListDelegatedAdministrators",
        "organizations:ListAWSServiceAccessForOrganization",
        "organizations:DescribeOrganizationalUnit",
        "organizations:DescribeAccount",
        "organizations:DescribeOrganization"
      ],
      "Resource": "*"
    }
  ]
}

```

Allow full access to all Amazon Inspector resources

This example shows a policy that allows full access to all Amazon Inspector resources.

```

{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "inspector2:*",
      "Resource": "*"
    },
    {
      "Effect": "Allow",
      "Action": "iam:CreateServiceLinkedRole",
      "Resource": "*",
      "Condition": {
        "StringLike": {
          "iam:AWSServiceName": "inspector2.amazonaws.com"
        }
      }
    }
  ],
  {
    "Effect": "Allow",
    "Action": [
      "organizations:EnableAWSServiceAccess",
      "organizations:RegisterDelegatedAdministrator",
      "organizations:ListDelegatedAdministrators",

```

```
        "organizations:ListAWSServiceAccessForOrganization",
        "organizations:DescribeOrganizationalUnit",
        "organizations:DescribeAccount",
        "organizations:DescribeOrganization"
    ],
    "Resource": "*"
}
]
```

AWS managed policies for Amazon Inspector

An AWS managed policy is a standalone policy that is created and administered by AWS. AWS managed policies are designed to provide permissions for many common use cases so that you can start assigning permissions to users, groups, and roles.

Keep in mind that AWS managed policies might not grant least-privilege permissions for your specific use cases because they're available for all AWS customers to use. We recommend that you reduce permissions further by defining [customer managed policies](#) that are specific to your use cases.

You cannot change the permissions defined in AWS managed policies. If AWS updates the permissions defined in an AWS managed policy, the update affects all principal identities (users, groups, and roles) that the policy is attached to. AWS is most likely to update an AWS managed policy when a new AWS service is launched or new API operations become available for existing services.

For more information, see [AWS managed policies](#) in the *IAM User Guide*.

AWS managed policy: AmazonInspector2FullAccess

You can attach the `AmazonInspector2FullAccess` policy to your IAM identities.

This policy grants administrative permissions that allow full access to Amazon Inspector.

Permissions details

This policy includes the following permissions.

- `inspector2` – Allows full access to Amazon Inspector functionality.
- `iam` – Allows Amazon Inspector to create the service-linked roles `AWSServiceRoleForAmazonInspector2` and `AWSServiceRoleForAmazonInspector2Agentless`.
`AWSServiceRoleForAmazonInspector2` is required for Amazon Inspector to perform operations such as retrieve information about your Amazon EC2 instances, Amazon ECR repositories, and container images. It's also required for Amazon Inspector to analyze your VPC network and describe accounts that are associated with your organization.
`AWSServiceRoleForAmazonInspector2Agentless` is required for Amazon Inspector to perform operations, such as retrieve information about your Amazon EC2 instances and Amazon EBS snapshots. It's also required to decrypt Amazon EBS snapshots that are encrypted with AWS KMS keys. For more information, see [Using service-linked roles for Amazon Inspector](#).
- `organizations` – Allows administrators to use Amazon Inspector for an organization in AWS Organizations. When you [activate trusted access](#) for Amazon Inspector in AWS Organizations, members of the delegated administrator account can manage settings and view findings across their organization.
- `codeguru-security` – Allows administrators to use Amazon Inspector to retrieve information code snippets and change encryption settings for code that CodeGuru Security stores. For more information, see [Encryption at rest for code in your findings](#).

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "AllowFullAccessToInspectorApis",
      "Effect": "Allow",
      "Action": "inspector2:*",
      "Resource": "*"
    },
    {
      "Sid": "AllowAccessToCodeGuruApis",
      "Effect": "Allow",
```

```

    "Action": [
      "codeguru-security:BatchGetFindings",
      "codeguru-security:GetAccountConfiguration"
    ],
    "Resource": "*"
  },
  {
    "Sid": "AllowAccessToCreateSlr",
    "Effect": "Allow",
    "Action": "iam:CreateServiceLinkedRole",
    "Resource": "*",
    "Condition": {
      "StringEquals": {
        "iam:AWSServiceName": [
          "agentless.inspector2.amazonaws.com",
          "inspector2.amazonaws.com"
        ]
      }
    }
  },
  {
    "Sid": "AllowAccessToOrganizationApis",
    "Effect": "Allow",
    "Action": [
      "organizations:EnableAWSServiceAccess",
      "organizations:RegisterDelegatedAdministrator",
      "organizations:ListDelegatedAdministrators",
      "organizations:ListAWSServiceAccessForOrganization",
      "organizations:DescribeOrganizationalUnit",
      "organizations:DescribeAccount",
      "organizations:DescribeOrganization"
    ],
    "Resource": "*"
  }
]
}

```

AWS managed policy: AmazonInspector2ReadOnlyAccess

You can attach the `AmazonInspector2ReadOnlyAccess` policy to your IAM identities.

This policy grants permissions that allow read-only access to Amazon Inspector.

Permissions details

This policy includes the following permissions.

- `inspector2` – Allows read-only access to Amazon Inspector functionality.
- `organizations` – Allows details about Amazon Inspector coverage for an organization in AWS Organizations to be viewed.
- `codeguru-security` – Allows code snippets to be retrieved from CodeGuru Security. Also allows encryption settings for your code stored in CodeGuru Security to be viewed.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "organizations:ListDelegatedAdministrators",
        "organizations:ListAWSServiceAccessForOrganization",
        "organizations:DescribeOrganizationalUnit",
        "organizations:DescribeAccount",
        "organizations:DescribeOrganization",
        "inspector2:BatchGet*",
        "inspector2:List*",
        "inspector2:Describe*",
        "inspector2:Get*",
        "inspector2:Search*",
        "codeguru-security:BatchGetFindings",
        "codeguru-security:GetAccountConfiguration"
      ],
      "Resource": "*"
    }
  ]
}
```

AWS managed policy: AmazonInspector2ManagedCisPolicy

You can attach the `AmazonInspector2ManagedCisPolicy` policy to your IAM entities. This policy should be attached to a role that grants permissions to your Amazon EC2 instances to run CIS scans of the instance. You can use an IAM role to manage temporary credentials for

applications that are running on an EC2 instance and making AWS CLI or AWS API requests. This is preferable to storing access keys within the EC2 instance. To assign an AWS role to an EC2 instance and make it available to all of its applications, you create an instance profile that is attached to the instance. An instance profile contains the role and enables programs that are running on the EC2 instance to get temporary credentials. For more information, see [Use an IAM role to grant permissions to applications running on Amazon EC2 instances](#) in the *IAM User Guide*.

Permissions details

This policy includes the following permissions.

- `inspector2` – Allows access to actions used to run CIS scans.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "inspector2:StartCisSession",
        "inspector2:StopCisSession",
        "inspector2:SendCisSessionTelemetry",
        "inspector2:SendCisSessionHealth"
      ],
      "Resource": "*"
    }
  ]
}
```

AWS managed policy: AmazonInspector2ServiceRolePolicy

You can't attach the `AmazonInspector2ServiceRolePolicy` policy to your IAM entities. This policy is attached to a service-linked role that allows Amazon Inspector to perform actions on your behalf. For more information, see [Using service-linked roles for Amazon Inspector](#).

AWS managed policy: AmazonInspector2AgentlessServiceRolePolicy

You can't attach the `AmazonInspector2AgentlessServiceRolePolicy` policy to your IAM entities. This policy is attached to a service-linked role that allows Amazon Inspector to perform actions on your behalf. For more information, see [Using service-linked roles for Amazon Inspector](#).

Amazon Inspector updates to AWS managed policies

View details about updates to AWS managed policies for Amazon Inspector since this service began tracking these changes. For automatic alerts about changes to this page, subscribe to the RSS feed on the Amazon Inspector [Document history](#) page.

Change	Description	Date
AmazonInspector2ServiceRolePolicy – Updates to an existing policy	Amazon Inspector has added new permissions that allow Amazon Inspector to return function tags in AWS Lambda.	July 31, 2024
AmazonInspector2FullAccess – Updates to an existing policy	Amazon Inspector has added permissions that allow Amazon Inspector to create the service-linked role <code>AWSServiceRoleForAmazonInspector2Agentless</code> . This allows users to perform agent-based scanning and agentless scanning when they enable Amazon Inspector.	April 24, 2024
AmazonInspector2ManagedCisPolicy – New policy	Amazon Inspector has added a new managed policy that you can use as part of an instance profile to allow CIS scans on an instance.	January 23, 2024
AmazonInspector2ServiceRolePolicy – Updates to an existing policy	Amazon Inspector has added new permissions that allow Amazon Inspector to start CIS scans on target instances.	January 23, 2024

Change	Description	Date
AmazonInspector2AgentlessServiceRolePolicy – New policy	Amazon Inspector has added a new service-linked role policy to allow agentless scanning of EC2 instance.	November 27, 2023
AmazonInspector2ReadOnlyAccess – Updates to an existing policy	Amazon Inspector has added new permissions that allow read-only users to retrieve vulnerability intelligence details for package vulnerability findings.	September 22, 2023
AmazonInspector2ServiceRolePolicy – Updates to an existing policy	Amazon Inspector has added new permissions that allow Amazon Inspector to scan network configurations of Amazon EC2 instances that are part of Elastic Load Balancing target groups.	August 31, 2023
AmazonInspector2ReadOnlyAccess – Updates to an existing policy	Amazon Inspector has added new permissions that allow read-only users to export Software Bill of Materials (SBOM) for their resources.	June 29, 2023
AmazonInspector2ReadOnlyAccess – Updates to an existing policy	Amazon Inspector has added new permissions that allow read-only users to retrieve details of encryption settings for Lambda code scanning findings for their account.	June 13, 2023

Change	Description	Date
AmazonInspector2FullAccess – Updates to an existing policy	Amazon Inspector has added new permissions that allow users configure a customer managed KMS key to encrypt code in findings from Lambda code scanning.	June 13, 2023
AmazonInspector2ReadOnlyAccess – Updates to an existing policy	Amazon Inspector has added new permissions that allow read-only users to retrieve details of Lambda code scanning status and findings for their account.	May 02, 2023
AmazonInspector2ServiceRolePolicy – Updates to an existing policy	Amazon Inspector has added new permissions that allow Amazon Inspector to create AWS CloudTrail service-linked channels in your account when you activate Lambda scanning. This allows Amazon Inspector to monitor CloudTrail events in your account.	April 30, 2023
AmazonInspector2FullAccess – Updates to an existing policy	Amazon Inspector has added new permissions that allow users to retrieve details of code vulnerability findings from Lambda code scanning.	April 21, 2023

Change	Description	Date
AmazonInspector2ServiceRolePolicy – Updates to an existing policy	Amazon Inspector has added new permissions that allow Amazon Inspector to send information to Amazon EC2 Systems Manager about the custom paths a customer has defined for Amazon EC2 deep inspection.	April 17, 2023
AmazonInspector2ServiceRolePolicy – Updates to an existing policy	Amazon Inspector has added new permissions that allow Amazon Inspector to create AWS CloudTrail service-linked channels in your account when you activate Lambda scanning. This allows Amazon Inspector to monitor CloudTrail events in your account.	April 30, 2023
AmazonInspector2ServiceRolePolicy – Updates to an existing policy	Amazon Inspector has added new permissions that allow Amazon Inspector to request scans of the developer code in AWS Lambda functions , and receive scan data from Amazon CodeGuru Security. Additionally, Amazon Inspector has added permissions to review IAM policies. Amazon Inspector uses this information to scan Lambda functions for code vulnerabilities.	February 28, 2023

Change	Description	Date
AmazonInspector2ServiceRolePolicy – Updates to an existing policy	Amazon Inspector has added a new statement that allows Amazon Inspector to retrieve information from CloudWatch about when an AWS Lambda function was last invoked. Amazon Inspector uses this information to focus scans on the Lambda functions in your environment that have been active in the last 90 days.	February 20, 2023
AmazonInspector2ServiceRolePolicy – Updates to an existing policy	Amazon Inspector has added a new statement that allows Amazon Inspector to retrieve information about AWS Lambda functions, including each layer version that is associated with each function. Amazon Inspector uses this information to scan Lambda functions for security vulnerabilities.	November 28, 2022

Change	Description	Date
AmazonInspector2ServiceRolePolicy – Updates to an existing policy	Amazon Inspector has added a new action to allow Amazon Inspector to describe SSM association executions. Additionally, Amazon Inspector has added additional resource scoping to allow Amazon Inspector to create, update, delete, and start SSM associations with AmazonInspector2 owned SSM documents.	August 31, 2022
AmazonInspector2ServiceRolePolicy Updates to an existing policy	Amazon Inspector has updated the resource scoping of the policy to allow Amazon Inspector to collect software inventory in other AWS partitions.	August 12, 2022
AmazonInspector2ServiceRolePolicy – Updates to an existing policy	Amazon Inspector has restructured the resource scoping of the actions allowing Amazon Inspector to create, delete, and update SSM associations.	August 10, 2022
AmazonInspector2ReadOnlyAccess – New policy	Amazon Inspector added a new policy to allow read-only access to Amazon Inspector functionality.	January 21, 2022

Change	Description	Date
AmazonInspector2FullAccess – New policy	Amazon Inspector added a new policy to allow full access to Amazon Inspector functionality.	November 29, 2021
AmazonInspector2ServiceRolePolicy – New policy	Amazon Inspector added a new policy to allow Amazon Inspector to perform actions in other services on your behalf.	November 29, 2021
Amazon Inspector started tracking changes	Amazon Inspector started tracking changes for its AWS managed policies.	November 29, 2021

Using service-linked roles for Amazon Inspector

Amazon Inspector uses an AWS Identity and Access Management (IAM) [service-linked role](#) named `AWSServiceRoleForAmazonInspector2`. This service-linked role is an IAM role that is linked directly to Amazon Inspector. It is predefined by Amazon Inspector and it includes all the permissions that Amazon Inspector requires to call other AWS services on your behalf.

A service-linked role makes setting up Amazon Inspector easier because you don't have to manually add the necessary permissions. Amazon Inspector defines the permissions of its service-linked role and, unless defined otherwise, only Amazon Inspector can assume the role. The defined permissions include the trust policy and the permissions policy, and that permissions policy cannot be attached to any other IAM entity.

You must configure permissions to allow an IAM entity (such as a group or role) to create, edit, or delete a service-linked role. For more information, see [Service-linked role permissions](#) in the *IAM User Guide*. You can delete a service-linked role only after deleting its related resources. This protects your Amazon Inspector resources because you can't inadvertently remove permission to access the resources.

For information about other services that support service-linked roles, see [AWS services that work with IAM](#) and look for the services that have **Yes** in the **Service-linked roles** column. Choose a **Yes** with a link to review the service-linked role documentation for that service.

Service-linked role permissions for Amazon Inspector

Amazon Inspector uses the service-linked role named `AWSServiceRoleForAmazonInspector2`. This service-linked role trusts the `inspector2.amazonaws.com` service to assume the role.

The permissions policy for the role, which is named `AmazonInspector2ServiceRolePolicy`, allows Amazon Inspector to perform tasks such as:

- Use Amazon Elastic Compute Cloud (Amazon EC2) actions to retrieve information about your instances and network paths.
- Use AWS Systems Manager actions to retrieve inventory from your Amazon EC2 instances, and to retrieve information about third-party packages from custom paths.
- Use the AWS Systems Manager `SendCommand` action to invoke CIS scans for target instances.
- Use Amazon Elastic Container Registry actions to retrieve information about your container images.
- Use AWS Lambda actions to retrieve information about your Lambda functions.
- Use AWS Organizations actions to describe associated accounts.
- Use CloudWatch actions to retrieve information about the last time your Lambda functions were invoked.
- Use select IAM actions to retrieve information about your IAM policies that could create security vulnerabilities in your Lambda code.
- Use CodeGuru Security actions to perform scans of the code in your Lambda functions. Amazon Inspector uses the following CodeGuru Security actions:
 - `codeguru-security:CreateScan` – Grants permission to create CodeGuru Security scan.
 - `codeguru-security:GetScan` – Grants permission to retrieve CodeGuru Security scan metadata.
 - `codeguru-security:ListFindings` – Grants permission to retrieve findings generated by CodeGuru Security.
 - `codeguru-security>DeleteScansByCategory` – Grants permission for CodeGuru Security to delete scans initiated by Amazon Inspector.
 - `codeguru-security:BatchGetFindings` – Grants permission to retrieve a batch of specific findings generated by CodeGuru Security.

- Use select Elastic Load Balancing actions to preform network scans of EC2 instances that are part of Elastic Load Balancing target groups.

The role is configured with the following permissions policy.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "TirosPolicy",
      "Effect": "Allow",
      "Action": [
        "directconnect:DescribeConnections",
        "directconnect:DescribeDirectConnectGatewayAssociations",
        "directconnect:DescribeDirectConnectGatewayAttachments",
        "directconnect:DescribeDirectConnectGateways",
        "directconnect:DescribeVirtualGateways",
        "directconnect:DescribeVirtualInterfaces",
        "ec2:DescribeAvailabilityZones",
        "ec2:DescribeCustomerGateways",
        "ec2:DescribeInstances",
        "ec2:DescribeInternetGateways",
        "ec2:DescribeManagedPrefixLists",
        "ec2:DescribeNatGateways",
        "ec2:DescribeNetworkAcls",
        "ec2:DescribeNetworkInterfaces",
        "ec2:DescribePrefixLists",
        "ec2:DescribeRegions",
        "ec2:DescribeRouteTables",
        "ec2:DescribeSecurityGroups",
        "ec2:DescribeSubnets",
        "ec2:DescribeTransitGatewayAttachments",
        "ec2:DescribeTransitGatewayConnects",
        "ec2:DescribeTransitGatewayPeeringAttachments",
        "ec2:DescribeTransitGatewayRouteTables",
        "ec2:DescribeTransitGatewayVpcAttachments",
        "ec2:DescribeTransitGateways",
        "ec2:DescribeVpcEndpointServiceConfigurations",
        "ec2:DescribeVpcEndpoints",
        "ec2:DescribeVpcPeeringConnections",
        "ec2:DescribeVpcs",
        "ec2:DescribeVpnConnections",
```

```

    "ec2:DescribeVpnGateways",
    "ec2:GetManagedPrefixListEntries",
    "ec2:GetTransitGatewayRouteTablePropagations",
    "ec2:SearchTransitGatewayRoutes",
    "elasticloadbalancing:DescribeListeners",
    "elasticloadbalancing:DescribeLoadBalancerAttributes",
    "elasticloadbalancing:DescribeLoadBalancers",
    "elasticloadbalancing:DescribeRules",
    "elasticloadbalancing:DescribeTags",
    "elasticloadbalancing:DescribeTargetGroups",
    "elasticloadbalancing:DescribeTargetGroupAttributes",
    "elasticloadbalancing:DescribeTargetHealth",
    "network-firewall:DescribeFirewall",
    "network-firewall:DescribeFirewallPolicy",
    "network-firewall:DescribeResourcePolicy",
    "network-firewall:DescribeRuleGroup",
    "network-firewall:ListFirewallPolicies",
    "network-firewall:ListFirewalls",
    "network-firewall:ListRuleGroups",
    "tiros:CreateQuery",
    "tiros:GetQueryAnswer"
  ],
  "Resource": [
    "*"
  ]
},
{
  "Sid": "PackageVulnerabilityScanning",
  "Effect": "Allow",
  "Action": [
    "ecr:BatchGetImage",
    "ecr:BatchGetRepositoryScanningConfiguration",
    "ecr:DescribeImages",
    "ecr:DescribeRegistry",
    "ecr:DescribeRepositories",
    "ecr:GetAuthorizationToken",
    "ecr:GetDownloadUrlForLayer",
    "ecr:GetRegistryScanningConfiguration",
    "ecr:ListImages",
    "ecr:PutRegistryScanningConfiguration",
    "organizations:DescribeAccount",
    "organizations:DescribeOrganization",
    "organizations:ListAccounts",
    "ssm:DescribeAssociation",

```

```

    "ssm:DescribeAssociationExecutions",
    "ssm:DescribeInstanceInformation",
    "ssm:ListAssociations",
    "ssm:ListResourceDataSync"
  ],
  "Resource": "*"
},
{
  "Sid": "LambdaPackageVulnerabilityScanning",
  "Effect": "Allow",
  "Action": [
    "lambda:ListFunctions",
    "lambda:GetFunction",
    "lambda:GetLayerVersion",
    "lambda:ListTags",
    "cloudwatch:GetMetricData"
  ],
  "Resource": "*"
},
{
  "Sid": "GatherInventory",
  "Effect": "Allow",
  "Action": [
    "ssm:CreateAssociation",
    "ssm:StartAssociationsOnce",
    "ssm>DeleteAssociation",
    "ssm:UpdateAssociation"
  ],
  "Resource": [
    "arn:aws:ec2:*:*:instance/*",
    "arn:aws:ssm:*:*:document/AmazonInspector2-*",
    "arn:aws:ssm:*:*:document/AWS-GatherSoftwareInventory",
    "arn:aws:ssm:*:*:managed-instance/*",
    "arn:aws:ssm:*:*:association/*"
  ]
},
{
  "Sid": "DataSyncCleanup",
  "Effect": "Allow",
  "Action": [
    "ssm:CreateResourceDataSync",
    "ssm>DeleteResourceDataSync"
  ],
  "Resource": [

```

```
    "arn:aws:ssm:*:*:resource-data-sync/InspectorResourceDataSync-do-not-delete"
  ]
},
{
  "Sid": "ManagedRules",
  "Effect": "Allow",
  "Action": [
    "events:PutRule",
    "events>DeleteRule",
    "events:DescribeRule",
    "events>ListTargetsByRule",
    "events:PutTargets",
    "events:RemoveTargets"
  ],
  "Resource": [
    "arn:aws:events:*:*:rule/DO-NOT-DELETE-AmazonInspector*ManagedRule"
  ]
},
{
  "Sid": "LambdaCodeVulnerabilityScanning",
  "Effect": "Allow",
  "Action": [
    "codeguru-security:CreateScan",
    "codeguru-security:GetAccountConfiguration",
    "codeguru-security:GetFindings",
    "codeguru-security:GetScan",
    "codeguru-security>ListFindings",
    "codeguru-security:BatchGetFindings",
    "codeguru-security>DeleteScansByCategory"
  ],
  "Resource": [
    "*"
  ]
},
{
  "Sid": "CodeGuruCodeVulnerabilityScanning",
  "Effect": "Allow",
  "Action": [
    "iam:GetRole",
    "iam:GetRolePolicy",
    "iam:GetPolicy",
    "iam:GetPolicyVersion",
    "iam>ListAttachedRolePolicies",
    "iam>ListPolicies",
```

```

    "iam:ListPolicyVersions",
    "iam:ListRolePolicies",
    "lambda:ListVersionsByFunction"
  ],
  "Resource": [
    "*"
  ],
  "Condition": {
    "ForAnyValue:StringEquals": {
      "aws:CalledVia": [
        "codeguru-security.amazonaws.com"
      ]
    }
  }
},
{
  "Sid": "Ec2DeepInspection",
  "Effect": "Allow",
  "Action": [
    "ssm:PutParameter",
    "ssm:GetParameters",
    "ssm>DeleteParameter"
  ],
  "Resource": [
    "arn:aws:ssm:*:*:parameter/inspector-aws/service/inspector-linux-application-paths"
  ],
  "Condition": {
    "StringEquals": {
      "aws:ResourceAccount": "${aws:PrincipalAccount}"
    }
  }
},
{
  "Sid": "AllowManagementOfServiceLinkedChannel",
  "Effect": "Allow",
  "Action": [
    "cloudtrail:CreateServiceLinkedChannel",
    "cloudtrail>DeleteServiceLinkedChannel"
  ],
  "Resource": [
    "arn:aws:cloudtrail:*:*:channel/aws-service-channel/inspector2/*"
  ],
  "Condition": {
    "StringEquals": {

```

```

    "aws:ResourceAccount": "${aws:PrincipalAccount}"
  }
}
},
{
  "Sid": "AllowListServiceLinkedChannels",
  "Effect": "Allow",
  "Action": [
    "cloudtrail:ListServiceLinkedChannels"
  ],
  "Resource": [
    "*"
  ],
  "Condition": {
    "StringEquals": {
      "aws:ResourceAccount": "${aws:PrincipalAccount}"
    }
  }
},
{
  "Sid": "AllowToRunInvokeCisSpecificDocuments",
  "Effect": "Allow",
  "Action": [
    "ssm:SendCommand",
    "ssm:GetCommandInvocation"
  ],
  "Resource": [
    "arn:aws:ssm:*:*:document/AmazonInspector2-InvokeInspectorSsmPluginCIS"
  ]
},
{
  "Sid": "AllowToRunCisCommandsToSpecificResources",
  "Effect": "Allow",
  "Action": [
    "ssm:SendCommand"
  ],
  "Resource": [
    "arn:aws:ec2:*:*:instance/*"
  ],
  "Condition": {
    "StringEquals": {
      "aws:ResourceAccount": "${aws:PrincipalAccount}"
    }
  }
}
}

```

```
},
{
  "Sid": "AllowToPutCloudwatchMetricData",
  "Effect": "Allow",
  "Action": [
    "cloudwatch:PutMetricData"
  ],
  "Resource": [
    "*"
  ],
  "Condition": {
    "StringEquals": {
      "cloudwatch:namespace": "AWS/Inspector2"
    }
  }
}
]
```

Creating a service-linked role for Amazon Inspector

You don't need to manually create a service-linked role. When you activate Amazon Inspector in the AWS Management Console, the AWS CLI, or the AWS API, Amazon Inspector creates the service-linked role for you.

Editing a service-linked role for Amazon Inspector

Amazon Inspector does not allow you to edit the `AWSServiceRoleForAmazonInspector2` service-linked role. After a service-linked role is created, you cannot change the name of the role because various entities might reference the role. However, you can edit the description of the role by using IAM. For more information, see [Editing a service-linked role](#) in the *IAM User Guide*.

Deleting a service-linked role for Amazon Inspector

If you no longer need to use Amazon Inspector, we recommend that you delete the `AWSServiceRoleForAmazonInspector2` service-linked role. Before you can delete the role, you must deactivate Amazon Inspector in each AWS Region where it's activated. When you deactivate Amazon Inspector, it doesn't delete the role for you. Therefore, if you activate Amazon Inspector again, it can use the existing role. That way you can avoid having an unused entity that's not actively monitored or maintained. However, you must clean up the resources for your service-linked role before you can manually delete it.

If you delete this service-linked role and then need to create it again, you can use the same process to re-create the role in your account. When you activate Amazon Inspector, Amazon Inspector re-creates the service-linked role for you.

Note

If the Amazon Inspector service is using the role when you try to delete the resources, the deletion might fail. If that happens, wait a few minutes and then try the operation again.

You can use the IAM console, the AWS CLI, or the AWS API to delete the `AWSServiceRoleForAmazonInspector2` service-linked role. For more information, see [Deleting a service-linked role](#) in the *IAM User Guide*.

Service-linked role permissions for Amazon Inspector agentless scans

Amazon Inspector agentless scanning uses the service-linked role named `AWSServiceRoleForAmazonInspector2Agentless`. This SLR allows Amazon Inspector to create an Amazon EBS volume snapshot in your account, and then access the data from that snapshot. This service-linked role trusts the `agentless.inspector2.amazonaws.com` service to assume the role.

Important

The statements in this service-linked role prevent Amazon Inspector from performing agentless scans on any EC2 instance that you have excluded from scans using the `InspectorEc2Exclusion` tag. Additionally the statements prevent Amazon Inspector from accessing encrypted data from a volume when the KMS key used to encrypt it has the `InspectorEc2Exclusion` tag. For more information, see [Excluding instances from Amazon Inspector scans](#).

The permissions policy for the role, which is named `AmazonInspector2AgentlessServiceRolePolicy`, allows Amazon Inspector to perform tasks such as:

- Use Amazon Elastic Compute Cloud (Amazon EC2) actions to retrieve information about your EC2 instances, volumes, and snapshots.

- Use Amazon EC2 tagging actions to tag snapshots for scans with the `InspectorScan` tag key.
- Use Amazon EC2 snapshot actions to create snapshots, tag them with the `InspectorScan` tag key, and then delete snapshots of Amazon EBS volumes that have been tagged with the `InspectorScan` tag key.
- Use Amazon EBS actions to retrieve information from snapshots tagged with the `InspectorScan` tag key.
- Use select AWS KMS decryption actions to decrypt snapshots encrypted with AWS KMS customer managed keys. Amazon Inspector does not decrypt snapshots when the KMS key used to encrypt them is tagged with the `InspectorEc2Exclusion` tag.

The role is configured with the following permissions policy.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "InstanceIdentification",
      "Effect": "Allow",
      "Action": [
        "ec2:DescribeInstances",
        "ec2:DescribeVolumes",
        "ec2:DescribeSnapshots"
      ],
      "Resource": "*"
    },
    {
      "Sid": "GetSnapshotData",
      "Effect": "Allow",
      "Action": [
        "ebs:ListSnapshotBlocks",
        "ebs:GetSnapshotBlock"
      ],
      "Resource": "arn:aws:ec2:*:*:snapshot/*",
      "Condition": {
        "StringLike": {
          "aws:ResourceTag/InspectorScan": "*"
        }
      }
    }
  ],
}
```

```

{
  "Sid": "CreateSnapshotsAnyInstanceOrVolume",
  "Effect": "Allow",
  "Action": "ec2:CreateSnapshots",
  "Resource": [
    "arn:aws:ec2:*:*:instance/*",
    "arn:aws:ec2:*:*:volume/*"
  ]
},
{
  "Sid": "DenyCreateSnapshotsOnExcludedInstances",
  "Effect": "Deny",
  "Action": "ec2:CreateSnapshots",
  "Resource": "arn:aws:ec2:*:*:instance/*",
  "Condition": {
    "StringEquals": {
      "ec2:ResourceTag/InspectorEc2Exclusion": "true"
    }
  }
},
{
  "Sid": "CreateSnapshotsOnAnySnapshotOnlyWithTag",
  "Effect": "Allow",
  "Action": "ec2:CreateSnapshots",
  "Resource": "arn:aws:ec2:*:*:snapshot/*",
  "Condition": {
    "Null": {
      "aws:TagKeys": "false"
    },
    "ForAllValues:StringEquals": {
      "aws:TagKeys": "InspectorScan"
    }
  }
},
{
  "Sid": "CreateOnlyInspectorScanTagOnlyUsingCreateSnapshots",
  "Effect": "Allow",
  "Action": "ec2:CreateTags",
  "Resource": "arn:aws:ec2:*:*:snapshot/*",
  "Condition": {
    "StringLike": {
      "ec2:CreateAction": "CreateSnapshots"
    },
    "Null": {

```

```

    "aws:TagKeys": "false"
  },
  "ForAllValues:StringEquals": {
    "aws:TagKeys": "InspectorScan"
  }
},
{
  "Sid": "DeleteOnlySnapshotsTaggedForScanning",
  "Effect": "Allow",
  "Action": "ec2:DeleteSnapshot",
  "Resource": "arn:aws:ec2:*:*:snapshot/*",
  "Condition": {
    "StringLike": {
      "ec2:ResourceTag/InspectorScan": "*"
    }
  }
},
{
  "Sid": "DenyKmsDecryptForExcludedKeys",
  "Effect": "Deny",
  "Action": "kms:Decrypt",
  "Resource": "arn:aws:kms:*:*:key/*",
  "Condition": {
    "StringEquals": {
      "aws:ResourceTag/InspectorEc2Exclusion": "true"
    }
  }
},
{
  "Sid": "DecryptSnapshotBlocksVolContext",
  "Effect": "Allow",
  "Action": "kms:Decrypt",
  "Resource": "arn:aws:kms:*:*:key/*",
  "Condition": {
    "StringEquals": {
      "aws:ResourceAccount": "${aws:PrincipalAccount}"
    },
    "StringLike": {
      "kms:ViaService": "ec2.*.amazonaws.com",
      "kms:EncryptionContext:aws:ebs:id": "vol-*"
    }
  }
},
},

```

```

{
  "Sid": "DecryptSnapshotBlocksSnapContext",
  "Effect": "Allow",
  "Action": "kms:Decrypt",
  "Resource": "arn:aws:kms:*:*:key/*",
  "Condition": {
    "StringEquals": {
      "aws:ResourceAccount": "${aws:PrincipalAccount}"
    },
    "StringLike": {
      "kms:ViaService": "ec2.*.amazonaws.com",
      "kms:EncryptionContext:aws:ebs:id": "snap-*"
    }
  }
},
{
  "Sid": "DescribeKeysForEbsOperations",
  "Effect": "Allow",
  "Action": "kms:DescribeKey",
  "Resource": "arn:aws:kms:*:*:key/*",
  "Condition": {
    "StringEquals": {
      "aws:ResourceAccount": "${aws:PrincipalAccount}"
    },
    "StringLike": {
      "kms:ViaService": "ec2.*.amazonaws.com"
    }
  }
},
{
  "Sid": "ListKeyResourceTags",
  "Effect": "Allow",
  "Action": "kms:ListResourceTags",
  "Resource": "arn:aws:kms:*:*:key/*"
}
]
}

```

Creating a service-linked role for agentless scanning

You don't need to manually create a service-linked role. When you activate Amazon Inspector in the AWS Management Console, the AWS CLI, or the AWS API, Amazon Inspector creates the service-linked role for you.

Editing a service-linked role for agentless scanning

Amazon Inspector does not allow you to edit the `AWSServiceRoleForAmazonInspector2Agentless` service-linked role. After a service-linked role is created, you cannot change the name of the role because various entities might reference the role. However, you can edit the description of the role by using IAM. For more information, see [Editing a service-linked role](#) in the *IAM User Guide*.

Deleting a service-linked role for agentless scanning

If you no longer need to use a feature or service that requires a service-linked role, we recommend that you delete that role. That way you don't have an unused entity that isn't actively monitored or maintained.

Important

In order to delete the `AWSServiceRoleForAmazonInspector2Agentless` role, you must set your scan mode to agent-based in all Regions where agentless scanning is available.

To manually delete the service-linked role using IAM

Use the IAM console, the AWS CLI, or the AWS API to delete the `AWSServiceRoleForAmazonInspector2Agentless` service-linked role. For more information, see [Deleting a service-linked role](#) in the *IAM User Guide*.

Troubleshooting Amazon Inspector identity and access

Use the following information to help you diagnose and fix common issues that you might encounter when working with Amazon Inspector and IAM.

Topics

- [I am not authorized to perform an action in Amazon Inspector](#)
- [I am not authorized to perform iam:PassRole](#)
- [I want to allow people outside of my AWS account to access my Amazon Inspector resources](#)

I am not authorized to perform an action in Amazon Inspector

If you receive an error that you're not authorized to perform an action, your policies must be updated to allow you to perform the action.

The following example error occurs when the `mateojackson` IAM user tries to use the console to view details about a fictional `my-example-widget` resource but doesn't have the fictional `inspector2:GetWidget` permissions.

```
User: arn:aws:iam::123456789012:user/mateojackson is not authorized to perform:
inspector2:GetWidget on resource: my-example-widget
```

In this case, the policy for the `mateojackson` user must be updated to allow access to the `my-example-widget` resource by using the `inspector2:GetWidget` action.

If you need help, contact your AWS administrator. Your administrator is the person who provided you with your sign-in credentials.

I am not authorized to perform iam:PassRole

If you receive an error that you're not authorized to perform the `iam:PassRole` action, your policies must be updated to allow you to pass a role to Amazon Inspector.

Some AWS services allow you to pass an existing role to that service instead of creating a new service role or service-linked role. To do this, you must have permissions to pass the role to the service.

The following example error occurs when an IAM user named `marymajor` tries to use the console to perform an action in Amazon Inspector. However, the action requires the service to have permissions that are granted by a service role. Mary does not have permissions to pass the role to the service.

```
User: arn:aws:iam::123456789012:user/marymajor is not authorized to perform:
iam:PassRole
```

In this case, Mary's policies must be updated to allow her to perform the `iam:PassRole` action.

If you need help, contact your AWS administrator. Your administrator is the person who provided you with your sign-in credentials.

I want to allow people outside of my AWS account to access my Amazon Inspector resources

You can create a role that users in other accounts or people outside of your organization can use to access your resources. You can specify who is trusted to assume the role. For services that support resource-based policies or access control lists (ACLs), you can use those policies to grant people access to your resources.

To learn more, consult the following:

- To learn whether Amazon Inspector supports these features, see [How Amazon Inspector works with IAM](#).
- To learn how to provide access to your resources across AWS accounts that you own, see [Providing access to an IAM user in another AWS account that you own](#) in the *IAM User Guide*.
- To learn how to provide access to your resources to third-party AWS accounts, see [Providing access to AWS accounts owned by third parties](#) in the *IAM User Guide*.
- To learn how to provide access through identity federation, see [Providing access to externally authenticated users \(identity federation\)](#) in the *IAM User Guide*.
- To learn the difference between using roles and resource-based policies for cross-account access, see [Cross account resource access in IAM](#) in the *IAM User Guide*.

Monitoring Amazon Inspector

Monitoring is an important part of maintaining the availability, reliability, and performance of Amazon Inspector and other AWS solutions. AWS provides tools to monitor Amazon Inspector, report issues that occur, and take actions to remediate these issues:

- [Amazon EventBridge](#) is an AWS service that uses events to connect application components together, making it easier for you to build scalable event-driven applications. EventBridge delivers a stream of real-time data from your applications, Software-as-a-Service (SaaS) applications, and AWS services and routes, so you can monitor events that happen in services and build event-driven architectures.
- [AWS CloudTrail](#) is an AWS service that captures API calls and related events made by or on behalf of your AWS account. CloudTrail delivers the log files to an Amazon S3 bucket that you specify, so you can identify which users and accounts called AWS, the source IP address from where calls were made, and when the calls occurred.

Logging Amazon Inspector API calls using AWS CloudTrail

Amazon Inspector is integrated with AWS CloudTrail, a service that provides a record of actions taken by an IAM user or role, or an AWS service, in Amazon Inspector. CloudTrail captures all API calls for Amazon Inspector as events. The calls captured include calls from the Amazon Inspector console and calls to the Amazon Inspector API operations. If you create a trail, you can enable continuous delivery of CloudTrail events to an Amazon S3 bucket, including events for Amazon Inspector. 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 Amazon Inspector.
- The IP address from which the request was made.
- Who made the request.
- When the request was made.

To learn more about CloudTrail, see the [AWS CloudTrail User Guide](#).

Amazon Inspector information in CloudTrail

CloudTrail is enabled on your AWS account when you create the account. When activity occurs in Amazon Inspector, 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 Amazon Inspector, 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 AWS 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 the following topics:

- [Overview for creating a trail](#)
- [CloudTrail supported services and integrations](#)
- [Configuring Amazon SNS notifications for CloudTrail](#)
- [Receiving CloudTrail log files from multiple accounts](#)
- [Receiving CloudTrail log files from multiple regions](#)

All Amazon Inspector actions are logged by CloudTrail. All actions that Amazon Inspector can make are documented in the [Amazon Inspector API Reference](#). For example, calls to the `CreateFindingsReport`, `ListCoverage`, and `UpdateOrganizationConfiguration` actions generate entries in the CloudTrail log files.

Every event or log entry contains information about who generated the request. The identity information helps you determine the following:

- Whether the request was made with root user or IAM user credentials.
- Whether the request was made with temporary security credentials for a role or a federated user.
- Whether the request was made by another AWS service.

For more information, see the [CloudTrail userIdentity element](#).

Understanding Amazon Inspector 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. Events include 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.

Amazon Inspector Scan information in CloudTrail

Amazon Inspector Scan is integrated with CloudTrail. All Amazon Inspector Scan API operations are logged as management events. For a list of the Amazon Inspector Scan API operations that Amazon Inspector logs to CloudTrail, see [Amazon Inspector Scan](#) in the Amazon Inspector API Reference.

The following example shows a CloudTrail log entry that demonstrates the `ScanSbom` action:

```
{
  "eventVersion": "1.08",
  "userIdentity": {
    "type": "AssumedRole",
    "principalId": "AROA123456789EXAMPLE:akua_mansa",
    "arn": "arn:aws:sts::111122223333:assumed-role/Admin/akua_mansa",
    "accountId": "111122223333",
    "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
```

```
    "sessionContext": {
      "sessionIssuer": {
        "type": "Role",
        "principalId": "AROAI23456789EXAMPLE",
        "arn": "arn:aws:iam::111122223333:role/Admin",
        "accountId": "111122223333",
        "userName": "Admin"
      },
      "webIdFederationData": {},
      "attributes": {
        "creationDate": "2023-10-17T15:22:59Z",
        "mfaAuthenticated": "false"
      }
    }
  },
  "eventTime": "2023-10-17T16:02:34Z",
  "eventSource": "gamma-inspector-scan.amazonaws.com",
  "eventName": "ScanSbom",
  "awsRegion": "us-east-1",
  "sourceIPAddress": "203.0.113.0",
  "userAgent": "aws-sdk-java/2.20.162 Mac_OS_X/13.5.2 OpenJDK_64-
Bit_Server_VM/17.0.8+7-LTS Java/17.0.8 vendor/Amazon.com_Inc. io/sync http/
URLConnection cfg/retry-mode/legacy",
  "requestParameters": {
    "sbom": {
      "specVersion": "1.5",
      "metadata": {
        "component": {
          "name": "debian",
          "type": "operating-system",
          "version": "9"
        }
      },
      "components": [
        {
          "name": "packageOne",
          "purl": "pkg:deb/debian/packageOne@1.0.0?arch=x86_64&distro=9",
          "type": "application"
        }
      ],
      "bomFormat": "CycloneDX"
    }
  },
  "responseElements": null,
```

```
"requestID": "f041a27f-f33e-4f70-b09b-5fbc5927282a",
"eventID": "abc8d1e4-d214-4f07-bc56-8a31be6e36fe",
"readOnly": true,
"eventType": "AwsApiCall",
"managementEvent": true,
"recipientAccountId": "111122223333",
"eventCategory": "Management"
}
```

Compliance validation for Amazon Inspector

To learn whether an AWS service is within the scope of specific compliance programs, see [AWS services in Scope by Compliance Program](#) and choose the compliance program that you are interested in. For general information, see [AWS Compliance Programs](#).

You can download third-party audit reports using AWS Artifact. For more information, see [Downloading Reports in AWS Artifact](#).

Your compliance responsibility when using AWS services is determined by the sensitivity of your data, your company's compliance objectives, and applicable laws and regulations. AWS provides the following resources to help with compliance:

- [Security and Compliance Quick Start Guides](#) – These deployment guides discuss architectural considerations and provide steps for deploying baseline environments on AWS that are security and compliance focused.
- [Architecting for HIPAA Security and Compliance on Amazon Web Services](#) – This whitepaper describes how companies can use AWS to create HIPAA-eligible applications.

Note

Not all AWS services are HIPAA eligible. For more information, see the [HIPAA Eligible Services Reference](#).

- [AWS Compliance Resources](#) – This collection of workbooks and guides might apply to your industry and location.
- [AWS Customer Compliance Guides](#) – Understand the shared responsibility model through the lens of compliance. The guides summarize the best practices for securing AWS services and map the guidance to security controls across multiple frameworks (including National Institute of

Standards and Technology (NIST), Payment Card Industry Security Standards Council (PCI), and International Organization for Standardization (ISO)).

- [Evaluating Resources with Rules](#) in the *AWS Config Developer Guide* – The AWS Config service assesses how well your resource configurations comply with internal practices, industry guidelines, and regulations.
- [AWS Security Hub](#) – This AWS service provides a comprehensive view of your security state within AWS. Security Hub uses security controls to evaluate your AWS resources and to check your compliance against security industry standards and best practices. For a list of supported services and controls, see [Security Hub controls reference](#).
- [Amazon GuardDuty](#) – This AWS service detects potential threats to your AWS accounts, workloads, containers, and data by monitoring your environment for suspicious and malicious activities. GuardDuty can help you address various compliance requirements, like PCI DSS, by meeting intrusion detection requirements mandated by certain compliance frameworks.
- [AWS Audit Manager](#) – This AWS service helps you continuously audit your AWS usage to simplify how you manage risk and compliance with regulations and industry standards.

Resilience in Amazon Inspector

The AWS global infrastructure is built around AWS Regions and Availability Zones. AWS Regions provide multiple, physically separated and isolated Availability Zones, which are connected to 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.

Infrastructure security in Amazon Inspector

As a managed service, Amazon Inspector is protected by AWS global network security. For information about AWS security services and how AWS protects infrastructure, see [AWS Cloud Security](#). To design your AWS environment using the best practices for infrastructure security, see [Infrastructure Protection](#) in *Security Pillar AWS Well-Architected Framework*.

You use AWS published API calls to access Amazon Inspector through the network. Clients must support the following:

- Transport Layer Security (TLS). We require TLS 1.2 and recommend TLS 1.3.

- Cipher suites with perfect forward secrecy (PFS) such as DHE (Ephemeral Diffie-Hellman) or ECDHE (Elliptic Curve Ephemeral Diffie-Hellman). Most modern systems such as Java 7 and later support these modes.

Additionally, requests must be signed by using an access key ID and a secret access key that is associated with an IAM principal. Or you can use the [AWS Security Token Service](#) (AWS STS) to generate temporary security credentials to sign requests.

Incident response in Amazon Inspector

Security is the highest priority at AWS. As mentioned in the [AWS shared responsibility model](#) under "Security of the Cloud," AWS is responsible for protecting the infrastructure running all of the services in the AWS Cloud. AWS is also responsible for any incident response associated with the Amazon Inspector service.

As an AWS customer, you share a responsibility for maintaining security in the AWS Cloud. This means you control the security you choose to implement, which includes all of the AWS tools and features you access. It also means you're responsible for incident response on your side of the shared responsibility model.

By establishing a security baseline that meets all of the objectives for your applications running in the AWS Cloud, you can detect deviations you can respond to. Because incident response is a complex topic, review the following resources to better understand the impact of incident response and how your choices might influence your corporate goals: [AWS Security Incident Response Guide](#), [AWS Security Best Practices](#), and [AWS Cloud Adoption Framework: Security Perspective](#).

Amazon Inspector integrations

Amazon Inspector integrates with other AWS services. These services can ingest data from Amazon Inspector, so you can view your findings in different ways. Review the following integration options to learn more.

Integrating Amazon Inspector with Amazon ECR

[Amazon Elastic Container Registry \(Amazon ECR\)](#) is an AWS-managed container image registry that supports private registries. Amazon ECR private registries host container images in a highly-available and scalable architecture. You can use Amazon Inspector to scan container images residing in your Amazon ECR repository for vulnerable operating system packages and programming language packages. For more information, see [Amazon Inspector integration with Amazon Elastic Container Registry \(Amazon ECR\)](#).

Amazon Inspector integration with AWS Security Hub

[AWS Security Hub](#) provides a comprehensive view of your security state in AWS and helps you check your environment against security industry standards and best practices. Security Hub collects security data from AWS accounts, services, and supported products. You can use Security Hub to ingest Amazon Inspector findings data and create a central location for findings in all of your integrated AWS services and AWS Partner Network products. For more information, see [Amazon Inspector integration with AWS Security Hub](#).

Amazon Inspector integration with Amazon Elastic Container Registry (Amazon ECR)

Amazon Elastic Container Registry is a fully managed container registry that supports Docker and OCI images and AWS artifacts. If you use Amazon ECR, you can activate [Enhanced Scanning](#) for your container registry. When you activate enhanced scanning, Amazon Inspector automatically detects and scans your container images for vulnerable operating system and programming language packages. This integration allows you to view Amazon Inspector findings for container images and manage the frequency and scope of scans in the Amazon ECR console. For more information, see [Scanning Amazon ECR container images with Amazon Inspector](#).

Activating the integration

You can activate the integration by activating Amazon Inspector scanning through the Amazon Inspector console or API, or by configuring your repository to use **Enhanced scanning** with Amazon Inspector through the Amazon ECR console or API.

For more information on activating the integration through Amazon Inspector, see [Automated scan types in Amazon Inspector](#).

For information on activating and configuring **Enhanced scanning** in Amazon ECR, see [Enhanced Scanning](#) in the Amazon ECR user guide.

Using the integration with a multi-account environment

If you are a member in a multi-account environment, you can activate enhanced scanning through Amazon ECR. However, once activated, it can only be deactivated by your Amazon Inspector delegated administrator. If it is deactivated, it reverts to basic scanning. For more information, see [Deactivating Amazon Inspector](#).

Amazon Inspector integration with AWS Security Hub

AWS Security Hub provides a comprehensive view of your security state in AWS and helps you check your environment against security industry standards and best practices. Security Hub collects security data from AWS accounts, services, and supported products. You can use the information Security Hub provides to analyze your security trends and identify the highest priority security issues. When you activate the integration, you can send findings from Amazon Inspector to Security Hub, and Security Hub can include these findings in its analysis of your security posture.

Security Hub tracks security issues as findings. Some of these findings can result from issues that other AWS services or third-party products detect. Security Hub uses a set of rules to detect security issues and generate findings. Security Hub provides tools that help you manage findings. Security Hub archives Amazon Inspector findings once the findings have been closed in Amazon Inspector. You can also [view a history of your findings and finding details](#), as well as [track the status of an investigation into a finding](#).

Security Hub findings use a standard JSON format called the [AWS Security Finding Format \(ASFF\)](#). The ASFF includes details about the source of the issue, the affected resources, and the current status of your findings.

Topics

- [Viewing Amazon Inspector findings in AWS Security Hub](#)
- [Activating and configuring the Amazon Inspector integration with Security Hub](#)
- [Disabling the flow of findings from an integration](#)
- [Viewing security controls for Amazon Inspector in Security Hub](#)

Viewing Amazon Inspector findings in AWS Security Hub

You can view Amazon Inspector Classic and Amazon Inspector findings in Security Hub.

Note

To filter on Amazon Inspector findings only, add "aws/inspector/ProductVersion": "2" to the filter bar. This filter excludes Amazon Inspector Classic findings from the Security Hub dashboard.

Example finding from Amazon Inspector

```
{
  "SchemaVersion": "2018-10-08",
  "Id": "arn:aws:inspector2:us-east-1:123456789012:finding/FINDING_ID",
  "ProductArn": "arn:aws:securityhub:us-east-1::product/aws/inspector",
  "ProductName": "Inspector",
  "CompanyName": "Amazon",
  "Region": "us-east-1",
  "GeneratorId": "AWSInspector",
  "AwsAccountId": "123456789012",
  "Types": [
    "Software and Configuration Checks/Vulnerabilities/CVE"
  ],
  "FirstObservedAt": "2023-01-31T20:25:38Z",
  "LastObservedAt": "2023-05-04T18:18:43Z",
  "CreatedAt": "2023-01-31T20:25:38Z",
  "UpdatedAt": "2023-05-04T18:18:43Z",
  "Severity": {
    "Label": "HIGH",
    "Normalized": 70
  },
  "Title": "CVE-2022-34918 - kernel",
```



```

"Description": "An issue was discovered in the Linux kernel through 5.18.9. A type
confusion bug in nft_set_elem_init (leading to a buffer overflow) could be used by a
local attacker to escalate privileges, a different vulnerability than CVE-2022-32250.
(The attacker can obtain root access, but must start with an unprivileged user
namespace to obtain CAP_NET_ADMIN access.) This can be fixed in nft_setelem_parse_data
in net/netfilter/nf_tables_api.c.",
"Remediation": {
  "Recommendation": {
    "Text": "Remediation is available. Please refer to the Fixed version in the
vulnerability details section above. For detailed remediation guidance for each of the
affected packages, refer to the vulnerabilities section of the detailed finding JSON."
  }
},
"ProductFields": {
  "aws/inspector/FindingStatus": "ACTIVE",
  "aws/inspector/inspectorScore": "7.8",
  "aws/inspector/resources/1/resourceDetails/awsEc2InstanceDetails/platform":
"AMAZON_LINUX_2",
  "aws/inspector/ProductVersion": "2",
  "aws/inspector/instanceId": "i-0f1ed287081bdf0fb",
  "aws/securityhub/FindingId": "arn:aws:securityhub:us-east-1::product/aws/inspector/
arn:aws:inspector2:us-east-1:123456789012:finding/FINDING_ID",
  "aws/securityhub/ProductName": "Inspector",
  "aws/securityhub/CompanyName": "Amazon"
},
"Resources": [
  {
    "Type": "AwsEc2Instance",
    "Id": "arn:aws:ec2:us-east-1:123456789012:i-0f1ed287081bdf0fb",
    "Partition": "aws",
    "Region": "us-east-1",
    "Tags": {
      "Patch Group": "SSM",
      "Name": "High-SEv-Test"
    }
  },
  "Details": {
    "AwsEc2Instance": {
      "Type": "t2.micro",
      "ImageId": "ami-0cff7528ff583bf9a",
      "IpV4Addresses": [
        "52.87.229.97",
        "172.31.57.162"
      ],
      "KeyName": "ACloudGuru",

```

```
    "IamInstanceProfileArn": "arn:aws:iam::123456789012:instance-profile/
AmazonSSMRoleForInstancesQuickSetup",
    "VpcId": "vpc-a0c2d7c7",
    "SubnetId": "subnet-9c934cb1",
    "LaunchedAt": "2022-07-26T21:49:46Z"
  }
}
],
"WorkflowState": "NEW",
"Workflow": {
  "Status": "NEW"
},
"RecordState": "ACTIVE",
"Vulnerabilities": [
  {
    "Id": "CVE-2022-34918",
    "VulnerablePackages": [
      {
        "Name": "kernel",
        "Version": "5.10.118",
        "Epoch": "0",
        "Release": "111.515.amzn2",
        "Architecture": "X86_64",
        "PackageManager": "OS",
        "FixedInVersion": "0:5.10.130-118.517.amzn2",
        "Remediation": "yum update kernel"
      }
    ],
    "Cvss": [
      {
        "Version": "2.0",
        "BaseScore": 7.2,
        "BaseVector": "AV:L/AC:L/Au:N/C:C/I:C/A:C",
        "Source": "NVD"
      },
      {
        "Version": "3.1",
        "BaseScore": 7.8,
        "BaseVector": "CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:H/A:H",
        "Source": "NVD"
      }
    ],
    {
      "Version": "3.1",
```

```

    "BaseScore": 7.8,
    "BaseVector": "CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:H/A:H",
    "Source": "NVD",
    "Adjustments": []
  }
],
"Vendor": {
  "Name": "NVD",
  "Url": "https://nvd.nist.gov/vuln/detail/CVE-2022-34918",
  "VendorSeverity": "HIGH",
  "VendorCreatedAt": "2022-07-04T21:15:00Z",
  "VendorUpdatedAt": "2022-10-26T17:05:00Z"
},
"ReferenceUrls": [
  "https://git.kernel.org/pub/scm/linux/kernel/git/netdev/net.git/commit/?id=7e6bc1f6cabcd30aba0b11219d8e01b952eacbb6",
  "https://lore.kernel.org/netfilter-devel/cd9428b6-7ffb-dd22-d949-d86f4869f452@randorisec.fr/T/",
  "https://www.debian.org/security/2022/dsa-5191"
],
"FixAvailable": "YES"
}
],
"FindingProviderFields": {
  "Severity": {
    "Label": "HIGH"
  },
  "Types": [
    "Software and Configuration Checks/Vulnerabilities/CVE"
  ]
},
"ProcessedAt": "2023-05-05T20:28:38.822Z"
}

```

Activating and configuring the Amazon Inspector integration with Security Hub

You can activate the Amazon Inspector integration with AWS Security Hub by [enabling Security Hub](#). After you enable Security Hub, the Amazon Inspector integration with AWS Security Hub is automatically activated, and Amazon Inspector begins sending all of its findings to Security Hub using the [AWS Security Finding Format \(ASFF\)](#).

Disabling the flow of findings from an integration

To stop Amazon Inspector from sending findings to Security Hub, you can use the Security Hub [console](#) or [API and AWS CLI](#).

Viewing security controls for Amazon Inspector in Security Hub

Security Hub analyzes findings from supported AWS and third-party products and runs automated and continuous security checks against rules to generate findings of its own. The rules are represented by security controls, which help you determine whether requirements in a standard are being met.

Amazon Inspector uses security controls to check whether Amazon Inspector features are or should be enabled. These features include the following:

- Amazon EC2 scanning
- Amazon ECR scanning
- Lambda standard scanning
- Lambda code scanning

For more information, see [Amazon Inspector controls](#) in the *AWS Security Hub User Guide*.

Supported operating systems and programming languages for Amazon Inspector

Amazon Inspector can scan software applications that are installed on the following:

- Amazon Elastic Compute Cloud (Amazon EC2) instances

Note

For Amazon EC2 instances, Amazon Inspector can scan for package vulnerabilities in operating systems that support agent-based scanning. Amazon Inspector can also scan for package vulnerabilities in operating systems and programming languages that support hybrid scanning. Amazon Inspector does not scan for toolchain vulnerabilities. The version of the programming language compiler used to build the application introduces these vulnerabilities.

- Container images stored in Amazon Elastic Container Registry (Amazon ECR) repositories

Note

For ECR container images, Amazon Inspector can scan for operating system and programming language package vulnerabilities. Amazon Inspector does not scan for toolchain vulnerabilities. The version of the programming language compiler used to build the application introduces these vulnerabilities.

- AWS Lambda functions

Note

For Lambda functions, Amazon Inspector can scan for programming language package vulnerabilities and code vulnerabilities. Amazon Inspector does not scan for toolchain vulnerabilities. The version of the programming language compiler used to build the application introduces these vulnerabilities.

When Amazon Inspector scans resources, Amazon Inspector sources more than 50 data feeds to generate findings for common vulnerabilities and exposures (CVEs). Examples of these sources

include vendor security advisories data feeds and threat intelligence feeds, as well as the National Vulnerability Database (NVD) and MITRE. Amazon Inspector updates vulnerability data from source feeds at least once daily.

For Amazon Inspector to scan a resource, the resource must be running a supported operating system or using a supported programming language. The topics in this section list the operating systems, programming languages, and runtimes Amazon Inspector supports for different resources and scan types. They also list discontinued operating systems.

Note

Amazon Inspector can provide only limited support for an operating system after a vendor discontinues support for the operating system.

Topics

- [Supported operating systems](#)
- [Discontinued operating systems](#)
- [Supported programming languages](#)
- [Supported runtimes](#)

Supported operating systems

This section lists the operating systems Amazon Inspector supports.

Supported operating systems: Amazon EC2 scanning

The following table lists the operating systems Amazon Inspector supports for the scanning of Amazon EC2 instances. It specifies the vendor security advisory for each operating system and which operating systems support [agent-based scanning](#) and [agentless scanning](#).

Note

When using the agent-based scanning method, you configure the SSM agent to perform continuous scans on all eligible instances. Amazon Inspector recommends that you configure a version of the SSM agent that's greater than 3.2.2086.0. For more information, see [Working with the SSM Agent](#) in the *Amazon EC2 Systems Manager User Guide*.

Linux operating system detections are supported only for the default package manager repository (rpm and dpkg) and don't include third-party applications, extended support repositories (BYOS RHEL, PAYG RHEL, and RHEL for SAP), and optional repositories (application streams).

Operating system	Version	Vendor security advisories	Agentless scan support	Agent-based scan support
AlmaLinux	8	ALSA	Yes	Yes
AlmaLinux	9	ALSA	Yes	Yes
Amazon Linux (AL2)	AL2	ALAS	Yes	Yes
Amazon Linux 2023 (AL2023)	AL2023	ALAS	Yes	Yes
Bottlerocket	1.7.0 and later	GHSA, CVE	No	Yes
Debian Server (Bullseye)	11	DSA	Yes	Yes
Debian Server (Bookworm)	12	DSA	Yes	Yes
Fedora	39	CVE	Yes	Yes
Fedora	40	CVE	Yes	Yes
OpenSUSE Leap	15.5	CVE	Yes	Yes
OpenSUSE Leap	15.6	CVE	Yes	Yes
Oracle Linux (Oracle)	7	ELSA	Yes	Yes
Oracle Linux (Oracle)	8	ELSA	Yes	Yes

Operating system	Version	Vendor security advisories	Agentless scan support	Agent-based scan support
Oracle Linux (Oracle)	9	ELSA	Yes	Yes
Red Hat Enterprise Linux (RHEL)	8	RHSA	Yes	Yes
Red Hat Enterprise Linux (RHEL)	9	RHSA	Yes	Yes
Rocky Linux	8	RLSA	Yes	Yes
Rocky Linux	9	RLSA	Yes	Yes
SUSE Linux Enterprise Server (SLES)	12.5	SUSE CVE	Yes	Yes
SUSE Linux Enterprise Server (SLES)	15.5	SUSE CVE	Yes	Yes
SUSE Linux Enterprise Server (SLES)	15.6	SUSE CVE	Yes	Yes
Ubuntu (Xenial)	16.04 (ESM)	USN, Ubuntu Pro	Yes	Yes
Ubuntu (Bionic)	18.04 (ESM)	USN, Ubuntu Pro	Yes	Yes
Ubuntu (Focal)	20.04 (LTS)	USN	Yes	Yes
Ubuntu (Jammy)	22.04 (LTS)	USN	Yes	Yes

Operating system	Version	Vendor security advisories	Agentless scan support	Agent-based scan support
Ubuntu (Noble Numbat)	24.04	USN	Yes	Yes
Windows Server	2016	MSKB	No	Yes
Windows Server	2019	MSKB	No	Yes
Windows Server	2022	MSKB	No	Yes
macOS (Mojave)	10.14	APPLE-SA	No	Yes
macOS (Catalina)	10.15	APPLE-SA	No	Yes
macOS (Big Sur)	11	APPLE-SA	No	Yes
macOS (Monterey)	12	APPLE-SA	No	Yes
macOS (Ventura)	13	APPLE-SA	No	Yes
macOS (Sonoma)	14	APPLE-SA	No	Yes

Supported operating systems: Amazon ECR scanning with Amazon Inspector

The following table lists the operating systems Amazon Inspector supports for the scanning of container images in Amazon ECR repositories. It also specifies the vendor security advisory for each operating system.

Operating system	Version	Vendor security advisories
Alpine Linux (Alpine)	3.17	Alpine SecDB

Operating system	Version	Vendor security advisories
Alpine Linux (Alpine)	3.18	Alpine SecDB
Alpine Linux (Alpine)	3.19	Alpine SecDB
Alpine Linux (Alpine)	3.20	Alpine SecDB
AlmaLinux	8	ALSA
AlmaLinux	9	ALSA
Amazon Linux (AL2)	AL2	ALAS
Amazon Linux 2023 (AL2023)	AL2023	ALAS
Debian Server (Bullseye)	11	DSA
Debian Server (Bookworm)	12	DSA
Fedora	39	CVE
Fedora	40	CVE
OpenSUSE Leap	15.5	CVE
OpenSUSE Leap	15.6	CVE
Oracle Linux (Oracle)	7	ELSA
Oracle Linux (Oracle)	8	ELSA
Oracle Linux (Oracle)	9	ELSA
Photon OS	4	PHSA
Photon OS	5	PHSA
Red Hat Enterprise Linux (RHEL)	8	RHSA

Operating system	Version	Vendor security advisories
Red Hat Enterprise Linux (RHEL)	9	RHSA
Rocky Linux	8	RLSA
Rocky Linux	9	RLSA
SUSE Linux Enterprise Server (SLES)	12.5	SUSE CVE
SUSE Linux Enterprise Server (SLES)	15.5	SUSE CVE
SUSE Linux Enterprise Server (SLES)	15.6	SUSE CVE
Ubuntu (Xenial)	16.04 (ESM)	USN, Ubuntu Pro
Ubuntu (Bionic)	18.04 (ESM)	USN, Ubuntu Pro
Ubuntu (Focal)	20.04 (LTS)	USN
Ubuntu (Jammy)	22.04 (LTS)	USN
Ubuntu (Noble Numbat)	24.04	USN

Supported operating systems: CIS scanning

The following table lists the operating systems Amazon Inspector supports for CIS scans. It also specifies the CIS benchmark version for each operating system.

Operating system	Version	CIS benchmark version
Amazon Linux 2	AL2	2.0.0
Amazon Linux 2023	AL2023	1.0.0

Operating system	Version	CIS benchmark version
Red Hat Enterprise Linux (RHEL)	8	3.0.0
Red Hat Enterprise Linux (RHEL)	9	1.0.0
Rocky Linux	8	2.0.0
Rocky Linux	9	1.0.0
Ubuntu (Bionic)	18.04 (LTS)	2.1.0
Ubuntu (Focal)	20.04 (LTS)	2.0.1
Ubuntu (Jammy)	22.04 (LTS)	1.0.0
Windows Server	2016	3.0.0
Windows Server	2019	2.0.0
Windows Server	2022	2.0.0

Discontinued operating systems

The following tables list which operating systems have been discontinued and when they were discontinued.

Even though Amazon Inspector doesn't provide full support for the following discontinued operating systems, Amazon Inspector continues to scan the Amazon EC2 instances and Amazon ECR container images running them. As a security best practice, we recommend moving to the supported version of a discontinued operating system. Findings that Amazon Inspector generates for a discontinued operating system should be used for informational purposes only.

In accordance with vendor policy, the following operating systems no longer receive patch updates. New security advisories might not be released for discontinued operating systems. Vendors can remove existing security advisories and detections from their feeds for operating systems that

reach the end of standard support. As a result, Amazon Inspector can stop generating findings for known CVEs.

Discontinued operating systems: Amazon EC2 scanning

Operating system	Version	Discontinued
Amazon Linux (AL1)	2012	December 31, 2021
CentOS Linux (CentOS)	7	June 30, 2024
CentOS Linux (CentOS)	8	December 31, 2021
Debian Server (Stretch)	9	June 30, 2022
Debian Server (Buster)	10	June 30, 2024
Fedora	35	December 13, 2022
Fedora	36	May 16, 2023
Fedora	37	December 15, 2023
Fedora	38	May 21, 2024
OpenSUSE Leap	15.2	December 1, 2021
OpenSUSE Leap	15.3	December 1, 2022
OpenSUSE Leap	15.4	December 7, 2023
Oracle Linux (Oracle)	6	March 1, 2021
Red Hat Enterprise Linux (RHEL)	7	June 30, 2024
SUSE Linux Enterprise Server (SLES)	12	June 30, 2016
SUSE Linux Enterprise Server (SLES)	12.1	May 31, 2017

Operating system	Version	Discontinued
SUSE Linux Enterprise Server (SLES)	12.2	March 31, 2018
SUSE Linux Enterprise Server (SLES)	12.3	June 30, 2019
SUSE Linux Enterprise Server (SLES)	12.4	June 30, 2020
SUSE Linux Enterprise Server (SLES)	15	December 31, 2019
SUSE Linux Enterprise Server (SLES)	15.1	January 31, 2021
SUSE Linux Enterprise Server (SLES)	15.2	December 31, 2021
SUSE Linux Enterprise Server (SLES)	15.3	December 31, 2022
SUSE Linux Enterprise Server (SLES)	15.4	December 31, 2023
Ubuntu (Trusty)	14.04 (ESM)	April 1, 2024
Ubuntu (Groovy)	20.10	July 22, 2021
Ubuntu (Hirsute)	21.04	January 20, 2022
Ubuntu (Impish)	21.10	July 31, 2022
Ubuntu (Kinetic)	22.10	July 20, 2023
Ubuntu (Lunar Lobster)	23.04	January 25, 2024
Ubuntu (Mantic Minotaur)	23.10	July 11, 2024
Windows Server	2012	October 10, 2023

Operating system	Version	Discontinued
Windows Server	2012 R2	October 10, 2023

Discontinued operating systems: Amazon ECR scanning

Operating system	Version	Discontinued
Alpine Linux (Alpine)	3.12	May 1, 2022
Alpine Linux (Alpine)	3.13	November 1, 2022
Alpine Linux (Alpine)	3.14	May 1, 2023
Alpine Linux (Alpine)	3.15	November 1, 2023
Alpine Linux (Alpine)	3.16	May 23, 2024
Amazon Linux (AL1)	2012	December 31, 2021
CentOS Linux (CentOS)	7	June 30, 2024
CentOS Linux (CentOS)	8	December 31, 2021
Debian Server (Stretch)	9	June 30, 2022
Debian Server (Buster)	10	June 30, 2024
Fedora	35	December 13, 2022
Fedora	36	May 16, 2023
Fedora	37	December 15, 2023
Fedora	38	May 21, 2024
OpenSUSE Leap	15.2	December 1, 2021
OpenSUSE Leap	15.3	December 1, 2022

Operating system	Version	Discontinued
OpenSUSE Leap	15.4	December 7, 2023
Oracle Linux (Oracle)	6	March 1, 2021
Photon OS	3	March 1, 2024
Red Hat Enterprise Linux (RHEL)	7	June 30, 2024
SUSE Linux Enterprise Server (SLES)	12	June 30, 2016
SUSE Linux Enterprise Server (SLES)	12.1	May 31, 2017
SUSE Linux Enterprise Server (SLES)	12.2	March 31, 2018
SUSE Linux Enterprise Server (SLES)	12.3	June 30, 2019
SUSE Linux Enterprise Server (SLES)	12.4	June 30, 2020
SUSE Linux Enterprise Server (SLES)	15	December 31, 2019
SUSE Linux Enterprise Server (SLES)	15.1	January 31, 2021
SUSE Linux Enterprise Server (SLES)	15.2	December 31, 2021
SUSE Linux Enterprise Server (SLES)	15.3	December 31, 2022
SUSE Linux Enterprise Server (SLES)	15.4	December 31, 2023

Operating system	Version	Discontinued
Ubuntu (Trusty)	14.04 (ESM)	April 1, 2024
Ubuntu (Groovy)	20.10	July 22, 2021
Ubuntu (Hirsute)	21.04	January 20, 2022
Ubuntu (Impish)	21.10	July 31, 2022
Ubuntu (Kinetic)	22.10	July 20, 2023
Ubuntu (Lunar Lobster)	23.04	January 25, 2024
Ubuntu (Mantic Minotaur)	23.10	July 11, 2024

Supported programming languages

This section lists the programming languages Amazon Inspector supports.

Supported programming languages: Amazon EC2 agentless scanning

Amazon Inspector currently supports the following programming languages when performing agentless scans on eligible Amazon EC2 instances. For more information, see [agentless scanning](#).

Note

Amazon Inspector doesn't scan for toolchain vulnerabilities in Go and Rust. The version of the programming language compiler used to build the application introduces these vulnerabilities.

- C#
- Go
- Java
- JavaScript
- PHP
- Python

- Ruby
- Rust

Supported programming languages: Amazon EC2 deep inspection

Amazon Inspector currently supports the following programming languages when performing deep inspection scans on Amazon EC2 Linux instances. For more information, see [Amazon Inspector deep inspection for Linux-based Amazon EC2 instances](#).

- Java (.ear, .jar, .par, and .war archive formats)
- JavaScript
- Python

Amazon Inspector uses Systems Manager Distributor to deploy the plugin for deep inspection of your Amazon EC2 instance.

Note

Deep inspection is not supported for Bottlerocket operating systems.

To perform deep inspection scans, Systems Manager Distributor and Amazon Inspector must support your Amazon EC2 instance operating system. For information about supported operating systems in Systems Manager Distributor, see [Supported package platforms and architectures](#) in the *Systems Manager User Guide*.

Supported programming languages: Amazon ECR scanning

Amazon Inspector currently supports the following programming languages when scanning container images in Amazon ECR repositories:

Note

Amazon Inspector doesn't scan for toolchain vulnerabilities in Go and Rust. The version of the programming language compiler used to build the application introduces these vulnerabilities.

- C#
- Go
- Java
- JavaScript
- PHP
- Python
- Ruby
- Rust

Supported runtimes

This section lists the runtimes Amazon Inspector supports.

Supported runtimes: Amazon Inspector Lambda standard scanning

Amazon Inspector Lambda standard scanning currently supports the following runtimes for the programming languages it can use when scanning Lambda functions for vulnerabilities in third-party software packages:

Note

Amazon Inspector doesn't scan for toolchain vulnerabilities in Go and Rust. The version of the programming language compiler used to build the application introduces these vulnerabilities.

- Go
 - go1.x
- Java
 - java8
 - java8.al2
 - java11
 - java17
 - java21

- .NET
 - .NET 6
 - .NET 8
- Node.js
 - nodejs12.x
 - nodejs14.x
 - nodejs16.x
 - nodejs18.x
 - nodejs20.x
- Python
 - python3.7
 - python3.8
 - python3.9
 - python3.10
 - python3.11
 - python3.12
- Ruby
 - ruby2.7
 - ruby3.2
 - ruby3.3
- Custom runtimes
 - AL2
 - AL2023

Supported runtimes: Amazon Inspector Lambda code scanning

Amazon Inspector Lambda code scanning currently supports the following runtimes for the programming languages it can use when scanning Lambda functions for vulnerabilities in code:

- Java
 - java8
 - java8.al2

- java11
- java17
- .NET
 - .NET 6
 - .NET 8
- Node.js
 - nodejs12.x
 - nodejs14.x
 - nodejs16.x
 - nodejs18.x
 - nodejs20.x
- Python
 - python3.7
 - python3.8
 - python3.9
 - python3.10
 - python3.11
 - python3.12
- Ruby
 - ruby2.7
 - ruby3.2
 - ruby3.3

Deactivating Amazon Inspector

You can deactivate Amazon Inspector in the Amazon Inspector console or with the Amazon Inspector API. If you deactivate all scan types for an account, Amazon Inspector is deactivated for that account automatically.

If you deactivate Amazon Inspector for an account, all scan types are deactivated for that account. Additionally, all Amazon Inspector scan settings, including filters, suppression rules, and findings are deleted for the account.

When you deactivate Amazon Inspector Amazon EC2 scanning, Amazon Inspector deletes the following SSM associations:

- `InspectorDistributor-do-not-delete`
- `InspectorInventoryCollection-do-not-delete`
- `InvokeInspectorSsmPlugin-do-not-delete`. Additionally, the Amazon Inspector SSM plugin installed through this association is removed from all of your Windows hosts. For more information, see [Scanning Windows EC2 instance](#).

Note

Once you deactivate Amazon Inspector, you no longer incur service charges. However, you can reactivate Amazon Inspector at any time.

For information about how to deactivate scan types for different resources, see [Deactivating a scan type](#).

Prerequisites

Depending on the account type, consider the following:

- If your account is a standalone Amazon Inspector account, you can deactivate Amazon Inspector at any time.
- If your account is a member account in a multi-account environment, you cannot deactivate Amazon Inspector. You must contact the delegated administrator for your organization to deactivate Amazon Inspector.

- If you're the delegated administrator for an organization, you must [disassociate all of your member accounts](#) before you deactivate Amazon Inspector.

Note

When you deactivate Amazon Inspector as the delegated administrator, you deactivate the auto-activate feature for your organization.

Deactivate Amazon Inspector

Note

Before you deactivate Amazon Inspector, consider [exporting your findings](#).

Console

To deactivate Amazon Inspector

1. Sign in using your credentials, and then open the Amazon Inspector console at <https://console.aws.amazon.com/inspector/v2/home>.
2. By using the AWS Region selector in the upper-right corner of the page, choose the Region in which you want to deactivate Amazon Inspector.
3. In the navigation pane, choose **General settings**.
4. Choose **Deactivate Inspector**.
5. When prompted for confirmation, enter **deactivate** in the text box, and then choose **Deactivate Inspector**.
6. (Recommended) Repeat these steps in each Region for which you want to deactivate Amazon Inspector.

API

Run the [Disable](#) API operation. In the request, provide the account IDs you are deactivating, and EC2, ECR, LAMBDA for resourceTypes to deactivate all scans, which will deactivate the account.

Amazon Inspector quotas

This section lists Amazon Inspector quotas per AWS Region.

Resource	Default	Comments
Member accounts	10,000	The maximum number of member accounts associated with an Amazon Inspector delegated administrator account. The limit is based on Quotas for AWS Organizations .
Suppression rules	500	The maximum number of saved suppression rules per AWS account per Region. You cannot request a quota increase.
Amazon EC2 network findings	10,000	The maximum number of Amazon EC2 network findings per AWS account. You cannot request a quota increase.
CIS scan configurations	500	The maximum number of CIS scan configurations. You cannot request a quota increase.

For a list of quotas associated with Amazon Inspector Classic, see [Amazon Inspector Classic service quotas](#) in the *AWS General Reference*. For a list of quotas associated with AWS Organizations, see [AWS Organizations service quotas](#) in the *AWS General Reference*.

Regions and endpoints

This topic includes tables that show endpoints for the Amazon Inspector Scan API and which AWS Regions support Amazon Inspector features.

To view the AWS Regions where Amazon Inspector is available, see [Amazon Inspector endpoints](#) in the *Amazon Web Services General Reference*.

Endpoints for Amazon Inspector Scan API

The following table shows the Regional endpoints that can be used when calling the [Amazon Inspector Scan API](#). When using the API you must supply the endpoint and its corresponding Region for the AWS Region you're currently authenticated to.

The naming convention for Amazon Inspector Scan endpoints is `inspector-scan.region.amazonaws.com`. For example, if you are authenticated in `us-west-2`, you would use the endpoint `inspector-scan.us-west-2.amazonaws.com` to call the `inspector-scan` API.

Region Name	Region	Endpoint	Protocol
US East (Ohio)	us-east-2	inspector-scan.us-east-2.amazonaws.com	HTTPS
		inspector-scan-fips.us-east-2.amazonaws.com	
US East (N. Virginia)	us-east-1	inspector-scan.us-east-1.amazonaws.com	HTTPS
		inspector-scan-fips.us-east-1.amazonaws.com	

Region Name	Region	Endpoint	Protocol
US West (N. California)	us-west-1	inspector-scan.us-west-1.amazonaws.com inspector-scan-fips.us-west-1.amazonaws.com	HTTPS
US West (Oregon)	us-west-2	inspector-scan.us-west-2.amazonaws.com inspector-scan-fips.us-west-2.amazonaws.com	HTTPS
Africa (Cape Town)	af-south-1	inspector-scan.af-south-1.amazonaws.com	HTTPS
Asia Pacific (Hong Kong)	ap-east-1	inspector-scan.ap-east-1.amazonaws.com	HTTPS
Asia Pacific (Jakarta)	ap-southeast-3	inspector-scan.ap-southeast-3.amazonaws.com	HTTPS
Asia Pacific (Mumbai)	ap-south-1	inspector-scan.ap-south-1.amazonaws.com	HTTPS
Asia Pacific (Osaka)	ap-northeast-3	inspector-scan.ap-northeast-3.amazonaws.com	HTTPS

Region Name	Region	Endpoint	Protocol
Asia Pacific (Seoul)	ap-northeast-2	inspector-scan.ap-northeast-2.amazonaws.com	HTTPS
Asia Pacific (Singapore)	ap-southeast-1	inspector-scan.ap-southeast-1.amazonaws.com	HTTPS
Asia Pacific (Sydney)	ap-southeast-2	inspector-scan.ap-southeast-2.amazonaws.com	HTTPS
Asia Pacific (Tokyo)	ap-northeast-1	inspector-scan.ap-northeast-1.amazonaws.com	HTTPS
Canada (Central)	ca-central-1	inspector-scan.ca-central-1.amazonaws.com	HTTPS
Europe (Frankfurt)	eu-central-1	inspector-scan.eu-central-1.amazonaws.com	HTTPS
Europe (Ireland)	eu-west-1	inspector-scan.eu-west-1.amazonaws.com	HTTPS
Europe (London)	eu-west-2	inspector-scan.eu-west-2.amazonaws.com	HTTPS
Europe (Milan)	eu-south-1	inspector-scan.eu-south-1.amazonaws.com	HTTPS

Region Name	Region	Endpoint	Protocol
Europe (Paris)	eu-west-3	inspector-scan.eu-west-3.amazonaws.com	HTTPS
Europe (Stockholm)	eu-north-1	inspector-scan.eu-north-1.amazonaws.com	HTTPS
Europe (Zurich)	eu-central-2	inspector-scan.eu-central-2.amazonaws.com	HTTPS
Middle East (Bahrain)	me-south-1	inspector-scan.me-south-1.amazonaws.com	HTTPS
South America (São Paulo)	sa-east-1	inspector-scan.sa-east-1.amazonaws.com	HTTPS
AWS GovCloud (US-East)	us-gov-east-1	inspector-scan.us-gov-east-1.amazonaws.com inspector-scan-fips.us-gov-east-1.amazonaws.com	HTTPS
AWS GovCloud (US-West)	us-gov-west-1	inspector-scan.us-gov-west-1.amazonaws.com inspector-scan-fips.us-gov-west-1.amazonaws.com	HTTPS

Region-specific feature availability

This section describes the availability of Amazon Inspector features by AWS Region.

Agentless EC2 scanning for Amazon EC2 Regions

The following table shows the AWS Regions where agentless scanning for Amazon EC2 is currently available.

Region Name	Region code
US East (N. Virginia)	us-east-1
US East (Ohio)	us-east-2
US West (N. California)	us-west-1
US West (Oregon)	us-west-2
Africa (Cape Town)	af-south-1
Asia Pacific (Hong Kong)	ap-east-1
Asia Pacific (Tokyo)	ap-northeast-1
Asia Pacific (Seoul)	ap-northeast-2
Asia Pacific (Osaka)	ap-northeast-3
Asia Pacific (Mumbai)	ap-south-1
Asia Pacific (Singapore)	ap-southeast-1
Asia Pacific (Sydney)	ap-southeast-2
Asia Pacific (Jakarta)	ap-southeast-3
Canada (Central)	ca-central-1
Europe (Stockholm)	eu-north-1
Europe (Frankfurt)	eu-central-1

Region Name	Region code
Europe (Zurich)	eu-central-2
Europe (Ireland)	eu-west-1
Europe (London)	eu-west-2
Europe (Paris)	eu-west-3
Europe (Milan)	eu-south-1
Middle East (Bahrain)	me-south-1
South America (São Paulo)	sa-east-1
AWS GovCloud (US-East)	us-gov-east-1
AWS GovCloud (US-West)	us-gov-west-1

Lambda code scanning Regions

The following table shows the AWS Regions where [Lambda code scanning](#) is currently available.

Region Name	Region code
US East (N. Virginia)	us-east-1
US West (Oregon)	us-west-2
US East (Ohio)	us-east-2
Asia Pacific (Sydney)	ap-southeast-2
Asia Pacific (Tokyo)	ap-northeast-1
Europe (Frankfurt)	eu-central-1
Europe (Ireland)	eu-west-1
Europe (London)	eu-west-2

Region Name	Region code
Europe (Stockholm)	eu-north-1
Asia Pacific (Singapore)	ap-southeast-1

⚠ Important

If you try to enable Lambda code scanning with the Amazon Inspector [Enable](#) API in an AWS Region where Lambda code scanning isn't available, you receive the following access denied error:

```
An error occurred (AccessDeniedException) when calling the Enable operation:  
Lambda code scanning is not supported in unsupported-AWS Region
```

AWS GovCloud (US) Regions

For the latest information, see [Amazon Inspector](#) in the *AWS GovCloud (US) User Guide*.

Document history

The following table describes important changes in each release of the *Amazon Inspector User Guide*, beginning in November 2021. To receive notifications about documentation updates, you can subscribe to an RSS feed.

Change	Description	Date
Updated functionality	Amazon Inspector adds AL2 as a supported runtime. For more information, see Supported operating systems and programming languages for Amazon Inspector .	August 26, 2024
Updated functionality	Amazon Inspector added a new statement to the AmazonInspector2ServiceRolePolicy policy. The new statement allows Amazon Inspector to return function tags in AWS Lambda.	July 31, 2024
Updated functionality	Amazon Inspector releases new security controls. For more information, see Amazon Inspector controls in the <i>AWS Security Hub User Guide</i> .	July 11, 2024
Updated functionality	The Amazon Inspector SBOM Generator now scans Dockerfiles and Docker container images for misconfigurations that can introduce security vulnerabilities. For more informati	June 10, 2024

on, see [Amazon Inspector Dockerfile checks](#).

[Updated functionality](#)

Amazon Inspector updates its [CI/CD integration feature](#) to support CodeCatalyst actions, so you can add Amazon Inspector vulnerability scans to your CodeCatalyst workflows. For more information, see [Using CodeCatalyst actions](#).

June 7, 2024

[Updated functionality](#)

Amazon Inspector includes an option to download a CSV file of CIS scan results. For more information, see [Viewing and downloading CIS scan results in Center for Internet Security \(CIS\) scans for Amazon EC2 instances](#).

May 3, 2024

[Updated functionality](#)

Amazon Inspector updates its [CI/CD integration feature](#) to support GitHub Actions, so you can add Amazon Inspector vulnerability scans to your GitHub workflows. For more information, see [Using Amazon Inspector with GitHub Actions](#).

April 29, 2024

Updated functionality	Amazon Inspector updates the managed policy AmazonInspector2FullAccess , so it creates the service-linked role AWSServiceRoleForAmazonInspector2Agentless . This allows users to perform agent-based scanning and agentless scanning when they enable Amazon Inspector.	April 24, 2024
Updated functionality	Amazon Inspector updates retention period for closed findings from 30 days to 7 days. For more information, see Understanding findings in Amazon Inspector .	February 12, 2024
Updated functionality	Amazon Inspector added a new statement to the AmazonInspector2ServiceRolePolicy policy. The new statement allows Amazon Inspector to start CIS scans for your instance.	January 23, 2024
New Policy	Amazon Inspector has added a new policy, AmazonInspector2ManagedCisPolicy policy, that you can use as part of in an instance profile to allow CIS scans on an instance.	January 23, 2024

[New Feature](#)

Amazon Inspector will now refresh the **ECR re-scan duration** of container images when you pull them. To change your re-scan duration based on push or pull dates see [Configuring the ECR re-scan duration](#).

January 23, 2024

[New Feature](#)

Amazon Inspector can now run Center for Internet Security (CIS) scans on EC2 instances. For more information, see [Amazon Inspector CIS scans](#).

January 23, 2024

[New Feature](#)

Amazon Inspector can now scan container images in your CI/CD pipelines. For more information, see [CI/CD integration with Amazon Inspector](#).

November 30, 2023

[New Policy](#)

Amazon Inspector has added a new policy that allows Amazon Inspector to scan Amazon EBS snapshots from your EC2 instance for agentless scanning. For more information on the policy, see [Agentless scanning](#).

November 27, 2023

New Feature	Amazon Inspector now supports scanning supported Linux Amazon EC2 instances without SSM agents through agentless scanning. For more information see Agentless scanning .	November 27, 2023
New supported resources	Amazon Inspector now supports scanning of MacOS Amazon EC2 instances. See Supported operating systems: Amazon EC2 scanning for supported MacOS versions.	October 5, 2023
New Regions	Amazon Inspector is now available in Asia Pacific (Jakarta), Africa (Cape Town), Asia Pacific (Osaka), and Europe (Zurich).	September 29, 2023
New feature	You can now exclude EC2 instances from Amazon Inspector scans using exclusion tags .	September 14, 2023
New feature	Amazon Inspector has added new permissions that allow Amazon Inspector to scan network configurations of Amazon EC2 instances that are part of Elastic Load Balancing target groups.	August 31, 2023

New feature	Amazon Inspector now provides vulnerability intelligence details for package vulnerability findings.	July 31, 2023
Updated functionality	Amazon Inspector has added new permissions that allow read-only users to export Software Bill of Materials (SBOM) for their resources.	June 29, 2023
New feature	You can now export SBOM for resources being scanned by Amazon Inspector.	June 13, 2023
New feature	Lambda code scanning is now generally available . New features have been added that allow you to encrypt code identified in your Lambda code scanning findings. Additionally Lambda code scanning now provides suggested remediation rewrites of your code.	June 13, 2023
Updated functionality	Amazon Inspector added a new statement to the AmazonInspector2ReadOnlyAccess policy . The new statements allows read-only users to retrieve details of Lambda code scanning status and findings for their account.	May 2, 2023

New feature

Amazon Inspector has added [Vulnerability database search](#) which allows you to check if Amazon Inspector covers a specific CVE.

May 1, 2023

Updated functionality

Amazon Inspector has added new permissions to the [AmazonInspector2ServiceRolePolicy policy](#) that allow Amazon Inspector to create AWS CloudTrail service-linked channels in your account when you activate Lambda scanning. This allows Amazon Inspector to monitor CloudTrail events in your account.

April 30, 2023

Updated functionality

Amazon Inspector added a new statement to the [AmazonInspector2FullAccess policy](#). The new statement allows users to retrieve details of code vulnerability findings from Lambda code scanning.

April 17, 2023

Updated functionality

Amazon Inspector added a new statement to the [AmazonInspector2ServiceRolePolicy](#) policy. The new statement allows Amazon Inspector to send information to Amazon EC2 Systems Manager about the custom paths you have defined for Amazon EC2 deep inspection.

April 17, 2023

New feature

Amazon Inspector adds additional support for Linux EC2 instances in the form of Amazon Inspector deep inspection, which scans your instances for package vulnerabilities in application programming language packages.

April 17, 2023

Updated functionality

Amazon Inspector added a new statement to the [AmazonInspector2ServiceRolePolicy](#) policy. The new statements allows Amazon Inspector to request scans of the developer code in AWS Lambda functions , and receive scan data from Amazon CodeGuru Security. Additionally Amazon Inspector has added permissions to review IAM policies. Amazon Inspector uses this information to scan Lambda functions for code vulnerabilities.

February 28, 2023

New feature

Amazon Inspector adds additional support for Lambda functions in the form of [Lambda code scanning](#), which scan the developer code of your Lambda functions for security vulnerabilities.

February 28, 2023

Updated functionality

Amazon Inspector added a new statement to the [AmazonInspector2ServiceRolePolicy](#) policy.

The new statement allows Amazon Inspector to retrieve information from CloudWatch about when an AWS Lambda function was last invoked. Amazon Inspector uses this information to focus scans on the Lambda functions in your environment that have been active in the last 90 days.

February 20, 2023

Updated functionality

Amazon Inspector added a new statement to the [AmazonInspector2ServiceRolePolicy](#) policy.

The new statement allows Amazon Inspector to retrieve information about your AWS Lambda functions. Amazon Inspector uses this information to scan your Lambda functions for security vulnerabilities.

November 28, 2022

New feature

Amazon Inspector adds support for [Scanning AWS Lambda functions](#).

November 28, 2022

Updated content

Added procedures, policy examples, and tips for [exporting findings reports](#) from Amazon Inspector to an Amazon Simple Storage Service (Amazon S3) bucket.

October 14, 2022

New content

Added information about [assessing Amazon Inspector coverage of your AWS environment](#) by using the Amazon Inspector console. The information includes descriptions of **Status** values for individual resources in your environment.

October 7, 2022

New feature

[Amazon Inspector now provides additional details about how to remediate package vulnerabilities](#). New fields have been added to finding details. The new fields provide context about whether a fix is available through a package update. If a fix is available, the **Suggested remediation** section of a finding shows the commands that you can run to make the fix.

September 2, 2022

Updated functionality

Amazon Inspector added a new action to the [AmazonInspector2ServiceRolePolicy policy](#). The new action allows Amazon Inspector to describe SSM association executions. Amazon Inspector also added additional resource scoping to allow Amazon Inspector to create, update, delete, and start SSM associations with AmazonInspector2 owned SSM documents.

August 31, 2022

New feature

[Amazon Inspector now supports scans for Windows instances](#). Amazon Inspector can now scan SSM managed instances running supported Windows operating systems. Scans of Windows hosts are performed by the Amazon Inspector SSM plugin, which is installed and invoked through new SSM associations automatically created by Amazon Inspector.

August 31, 2022

Updated functionality

Amazon Inspector updated the resource scoping of the [AmazonInspector2ServiceRolePolicy policy](#) to allow Amazon Inspector to collect software inventory in other AWS partitions.

August 12, 2022

Updated functionality

In the [AmazonInspector2ServiceRolePolicy](#) policy, Amazon Inspector restructured the resource scoping of the actions allowing Amazon Inspector to create, delete, and update SSM associations.

August 10, 2022

New feature

[Amazon Inspector now supports changing your ECR automated re-scan duration setting](#). The Amazon ECR automated re-scan duration setting determines how long Amazon Inspector continuously monitors images pushed into repositories. When an image is older than the scan duration, Amazon Inspector will no longer scan the image and close all existing findings for it. All new accounts will automatically have their ECR automated re-scan duration set to lifetime. Previously created accounts had an ECR automated re-scan duration of 30 days, but you can now choose from 30-day, 180-day, or lifetime durations for scans.

June 25, 2022

[New functionality](#)

Amazon Inspector added a new AWS managed policy, the [AmazonInspector2ReadOnlyAccess policy](#), to allow read-only access to Amazon Inspector functionality.

January 21, 2022

[General availability](#)

This is the initial public release of the *Amazon Inspector User Guide*.

November 29, 2021

AWS Glossary

For the latest AWS terminology, see the [AWS glossary](#) in the *AWS Glossary Reference*.