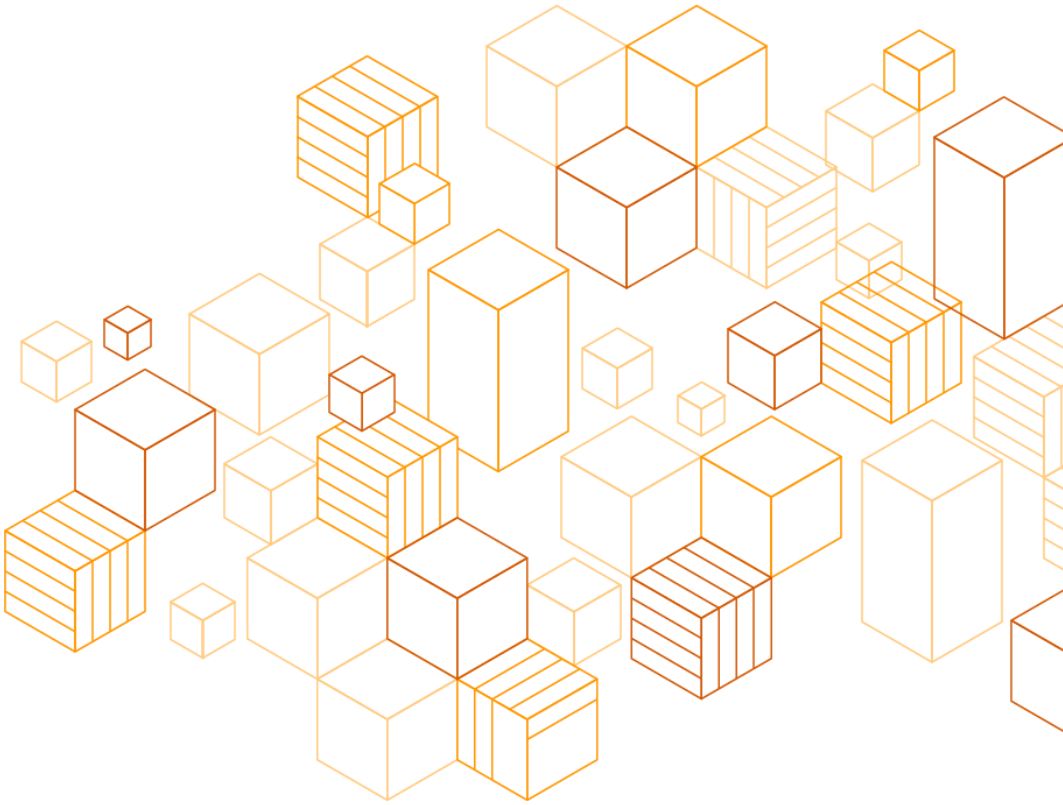
Security in Amazon CodeGuru Profiler

Published May 4, 2021



Notices

Customers are responsible for making their own independent assessment of the information in this document. This document: (a) is for informational purposes only, (b) represents current AWS product offerings and practices, which are subject to change without notice, and (c) does not create any commitments or assurances from AWS and its affiliates, suppliers or licensors. AWS products or services are provided “as is” without warranties, representations, or conditions of any kind, whether express or implied. The responsibilities and liabilities of AWS to its customers are controlled by AWS agreements, and this document is not part of, nor does it modify, any agreement between AWS and its customers.

Contents

About this Guide

This paper presents a deep dive into Amazon CodeGuru through a security lens. It provides a well-rounded picture of the service for new adopters, and a deeper understanding of CodeGuru for current users.

The intended audience includes Chief Information Security Officers (CISOs), information security groups, security analysts, enterprise architects, compliance teams, and those interested in understanding the security features of Amazon CodeGuru Profiler and its related services.

Introduction

Amazon CodeGuru is a developer tool that provides recommendations to improve your code quality and identify an application's most expensive lines of code. Customers can integrate CodeGuru into existing software development workflows to automate code reviews during application development, continuously monitor application performance in production, provide recommendations and visual clues for improving code quality and application performance, and reduce overall cost.

Amazon CodeGuru Reviewer uses machine learning to identify critical issues, security vulnerabilities, and hard-to-find bugs during application development to improve code quality.

Amazon CodeGuru Profiler provides recommendations about an application's most expensive lines of code by helping developers understand application runtime behavior, identify and remove code inefficiencies, improve performance, and significantly decrease compute costs.

The high-level flow for the profiler application is:

1. The profiler agent is instantiated (either within your application or via the CLI)
2. The agent begins sampling data to send to the service at five-minute intervals
3. CodeGuru profiler analyzes this data to generate visualizations and actionable recommendations

Shared Responsibility Model

Cloud security at AWS is the highest priority. As an AWS customer, you benefit from a data center and network architecture that is built to meet the requirements of the most security-sensitive organizations. Security is a shared responsibility between AWS and you. The AWS Shared Responsibility Model describes what customers and AWS do to architect for security and what AWS does to secure the cloud.

Like all services in AWS, CodeGuru operates on this Shared Responsibility model for security and compliance. CodeGuru is built using best practices on top of AWS technologies and it provides customers the tools they need to secure their data. The amount of security configuration required varies depending on which services you select and your data sensitivity.

The following sections outline how the AWS Shared Responsibility model covers CodeGuru, and explains associated customer and AWS responsibilities.

AWS Infrastructure Security

As a managed service, CodeGuru is protected by the AWS global network security procedures that are described in the Introduction to AWS Security whitepaper.

AWS is responsible for protecting the global infrastructure that runs the services offered in the AWS Cloud. This infrastructure comprises the hardware, software, networking, and facilities that run AWS services. Protecting this infrastructure is the number one priority of AWS. Customers can benefit from third party audits that verify compliance with industry standards and frameworks. For more information, visit AWS Compliance.

As a managed service, AWS is responsible for managing the underlying CodeGuru service. This includes patching and maintenance of the underlying compute functions, managing service availability, and data encryption in transit and at rest.

Amazon CodeGuru Profiler Service Security

The CodeGuru Profiler service includes a number of mechanisms that are designed to protect customer data. To review relevant outputs from compliance programs related to CodeGuru Profiler see AWS Artifact. To review relevant outputs from compliance programs related to CodeGuru Profiler, see AWS Artifact. If you have a request for a specific compliance program, contact aws-custfeedback-codeguru@amazon.com.

Data Captured

The CodeGuru Profiler agent is responsible for collecting data from a local instance, and sending it to the service endpoint for analysis. It does this by using language appropriate mechanisms in either the Java Virtual Machine (JVM), or through Python interfaces.

The profiler agent does not collect or store your applications source code in any way. There are only two types of data the tool collects to send to the profiler service – stack traces and heap summaries.

Stack traces

A stack trace is a sequence of function or method names in execution, followed by the function or method names that called them successively, continuing to the root of the

service process. The profiling agent does not have access to the names or values of function parameters, or the values of variables or application data. The stack trace data that is sent to the service includes only those function or method names and the profiling statistics that indicate their usage, which is used by the service to provide profiling group statistics such as CPU and latency data, and also to detect anomalies and recommendations on how to improve your application performance.

Heap summary (JVM only)

Heap memory is allocated to JVM and is shared by all threads in the application. The profiling agent can collect information about objects in the heap over time and securely share this with the service for analysis. Heap summary offers a consolidated view of memory use per object type (e.g. String, int, char[]), and custom types, during a given time frame (usually five minutes). CodeGuru tracks both the summed-up sizes of objects and their count. The data is sampled approximately in line with when a full garbage collection is done, and is sent to the CodeGuru Profiler service every 5 minutes. The collection of these statistics can be enabled or disabled with arguments provided to the agent. See Setting up CodeGuru Profiler (step 4) for instructions.

Encryption of data at rest

As data is collected by the CodeGuru Profiler profiling agent, it is sent to the back end service. Once received, it is stored in Amazon Simple Storage Service (Amazon S3), Amazon Kinesis, and Amazon DynamoDB when at rest within the service. The implementations of these services for the CodeGuru Profiler service all make use of their data-at-rest encryption capabilities.

Encryption of data in transit

When the profiling agent communicates with the CodeGuru Profiler service, all communication is secured with TLS connections and signed with the Signature Version 4 signing process. This is true for all downstream dependencies that the CodeGuru service calls to provide user functionality. All CodeGuru Profiler endpoints are secured with SHA-256 certificates, managed by AWS Certificate Manager Private Certificate Authority.

Data retention

When data is received in the CodeGuru Profiler service, it is aggregated in to profiles that represent 5-minute periods for each profiling group. A profiling group is designed to

represent a single application that uses the service. These statistics are then aggregated in to hourly and daily profiles. Retention periods for these profiles are:

- Five minutes – 14 days
- Hourly – 60 days
- Daily – 3 years

Data for a profiling group can be removed by deleting the profiling group. This deletes all recommendations reports within the deleted group. See Deleting a profiling group in the CodeGuru Profiler documentation for instructions.

Customer Configurable Security in CodeGuru Profiler

CodeGuru Profiler operates as a fully managed service in the AWS cloud, which exists outside of a customer network or Amazon Virtual Private Cloud (Amazon VPC). The profiling agent will operate within your compute instances - such as Amazon Elastic Compute Cloud (Amazon EC2) instances, containers running in Amazon Elastic Container Service (Amazon ECS) or Amazon Elastic Kubernetes Service (Amazon EKS), or serverless functions in AWS Lambda. The profiling agent can also run where ever your applications are running, including on-premises environments.

Beyond the specific security controls and options for the CodeGuru Profiler service within AWS, you should also ensure that appropriate network security controls are implemented to meet your specific security requirements that relate to how the agents can communicate with the CodeGuru service. These could include controls such as the use of VPC endpoints, firewalls or security groups, and other such mechanisms specific to your use cases and network configuration.

Network Security using VPC Endpoints

An IT security best practice is to minimize external exposure of applications so that you reduce the points where misconfiguration or unintended use can occur. You can do this with CodeGuru by using VPC endpoints when you call Amazon CodeGuru Profiler APIs. This allows you to integrate CodeGuru without needing to provide a direct path to the internet by limiting traffic to only within customer managed VPCs.

You establish a private connection between your VPC and CodeGuru Profiler by creating an interface VPC endpoint. These are powered by AWS PrivateLink, a

technology that enables you to privately access CodeGuru Profiler APIs without an internet gateway, NAT device, VPN connection, or AWS Direct Connect connection. Instances in your VPC do not require public IP addresses to communicate with CodeGuru Profiler APIs and traffic between your VPC and CodeGuru Profiler does not leave the AWS network. For more information, see VPC Endpoints in the CodeGuru documentation.

Considerations for using the profiler agent on-premises

Customers can choose to enable the CodeGuru Profiler service to instrument applications running outside of their AWS accounts. In this model, the agent will run in your on-premises servers and will continue to post data to the AWS account where your CodeGuru profiling group exists. This means you should consider networking between your application servers and AWS, as well as how credentials for the app servers will authenticate to the CodeGuru profiler service.

Networking from on-premises to the CodeGuru Profiler service

The CodeGuru profiling agent will still communicate to the service endpoints in AWS to share the same data as a workload running in AWS. This means that your application servers need to be able to talk the service endpoints (see Amazon CodeGuru Profiler endpoints and quotas for a full list). This can be achieved in a number of ways:

- **Direct internet access** – If your server can communicate to the internet, it can resolve the IP address of the appropriate endpoint and securely send information to the service.
- **Leveraging AWS Direct Connect** – Direct Connect allows you to establish private connectivity to an AWS region from your on-premises network. Within this service you can create a public virtual interface that will allow communication with AWS services over your private Direct Connect connection instead of traversing the internet.
- **Virtual Private Network Connection** - Leveraging a VPN allows connectivity between your on-premises network and AWS VPC. In this scenario you can leverage VPC Endpoints and have traffic routed over your VPN in to the VPC to allow the connectivity.

In all of these options, HTTPS connections are used to ensure secure communication between your application servers and the CodeGuru Profiler service endpoint.

Credentials for on-premises servers to authenticate to CodeGuru

When running in an AWS environment that has an IAM role associated with it, the CodeGuru profiler will use that role to get temporary credentials to communicate with the service. In an on-premises environment you must provide valid credentials that have permission to submit data to the service. When starting the agent, you can specify the `credentialPath` parameter to tell the profiler where to retrieve credentials from. See Enabling the agent from the command line for examples and see Configuration and credential file settings for guidance on managing credential files.

Using AWS Config for Security

AWS Config is a service that enables you to assess, audit, and evaluate the configurations of your AWS resources. Config continuously monitors and records your AWS resource configurations and provides you with the ability to define rules for provisioning and configuring AWS resources.

Currently there are no AWS Config Managed Rules for events related to CodeGuru. You can, however, develop rules with the AWS Config Rule Development Kit (RDK). For example, you could create a rule to track and alert on CodeGuru Profiler operations (including create, update, and delete), as well as changes to related Identity and Access Management (IAM) policies.

Access management

The CodeGuru Profiler service is integrated with IAM to allow for fine grained permissions control for CodeGuru Profiler resources. IAM provides two policy types for resource access authorization:

- **Resource policies** are appropriate in cases where the user creates resources and wants to allow other users to access those resources. The policy is attached directly to the resource and describes who can do what with the resource. The user is in control of the resource, and you can provide an IAM user with explicit access to a resource. The root AWS account always has access to manage resource policies, and is the owner of all resources created in that account. Alternatively, you can grant users explicit access to manage resource permissions.

- **Identity-based policies** are often used to enforce company-wide access policies. Capability policies are assigned to an IAM user either directly or indirectly using an IAM group. They can also be assigned to a role that will be assumed at run time. Capability policies define what capabilities (actions) the user can perform. These policies can override resource-based policy permissions by explicitly denying them.

CodeGuru Profiler supports both Profiling Groups policies, which is currently the only resource type provided by the service. Resource policies can only be used to grant permissions to send profiling data to the CodeGuru Profiler service from the profiling agent. The resource policies can be applied to a Profiling Group by selecting the “Give access to users and roles” option in the console or via the `PutPermissions` API and are scoped to the given Profiling Group.

Identity-based policies can be applied to any IAM user or role and can be used to allow access to many actions, including but not limited to: viewing and modifying Profiling Groups, applying and removing tags, and adding and removing resource policies. See Audience in CodeGuru Profiler for some examples.

CodeGuru Profiler provides two managed policies:

- AmazonCodeGuruProfilerReadOnlyAccess
- AmazonCodeGuruProfilerFullAccess

These policies allow users to get started quickly but may result in a broader than necessary permissions scope. To create IAM policies, follow the security best practice of granting least privilege, or granting only the permissions required to perform a task. Determine what users (and roles) need to do, then craft policies that allow them to perform only those tasks.

Policy best practices

Identity-based policies are powerful. They determine whether a user can create, access, or delete CodeGuru Profiler resources in your account, which can incur costs. When you create or edit identity-based policies, follow these guidelines and recommendations:

- **Get started using AWS managed policies** – To start using CodeGuru Profiler quickly, use AWS managed policies to give your employees the permissions they need. These policies are already available in your account and are maintained and updated by AWS. For more information, see Get started using permissions with AWS managed policies in the IAM User Guide.

- **Grant least privilege** – When you create custom policies, grant only the permissions required to perform a task. Start with a minimum set of permissions and grant additional permissions as necessary. Tools such as the AWS IAM Access Analyzer can help you ensure you are only providing access levels you intend. For more information, see Grant least privilege in the IAM User Guide.
- **Enable MFA for sensitive operations** – For extra security, require that IAM users use multi-factor authentication (MFA) to access sensitive resources or API operations. Additionally, you can centralize identities in to a Single Sign-On (SSO) solution ensures that access is managed in a single system, simplifying administration tasks such as enabling MFA. For more information, see Using multi-factor authentication (MFA) in AWS in the IAM User Guide.
- **Use policy conditions for extra security** – If practical, define the conditions under which your identity-based policies allow access to a resource. For example, you can write conditions to specify a range of allowable IP addresses that a request must come from. You can also write conditions to only allow requests within a specified date/time range, or to require the use of SSL or MFA. For more information, see IAM JSON policy elements: Condition in the IAM User Guide.

Using identity-based policies for CodeGuru Profiler

Permissions required to use the CodeGuru Profiler console

CodeGuru Profiler console users require a minimum set of permissions to describe other AWS resources for the AWS account. This includes permissions from the following services:

- CodeGuru Profiler
- ListUsers and ListRoles from the IAM service

> **NOTE:** If you create an IAM policy that is more restrictive than the minimum required permissions, the console won't function as intended.

Permissions required by the CodeGuru Profiler profiling agent

The CodeGuru Profiler profiling agent is imported into your profiled application. When your application runs, the agent starts in a different thread to profile your application. The following permissions are required to submit data to CodeGuru Profiler:

- codeguru-profiler:ConfigureAgent
- codeguru-profiler:PostAgentProfile

For more information, see Enabling the agent with code.

The following example policy grants the current AWS user permission to write to a single profiling group in the current AWS Region. Limiting access to individual resources provides a more secure approach when allowing the agent access to post data, as it ensures data can only be sent where you intend.

```
{"Statement": [{"Effect": "Allow",
      "Action": [
         "codeguru-profiler:ConfigureAgent",
         "codeguru-profiler:PostAgentProfile"
      ],
      "Resource": "arn:aws:codeguru-
profiler:<region>:<accountID>:profilingGroup/profilingGroupName"
   }]
}
```

Permissions required to access CodeGuru Profiler data

Data collected and submitted to CodeGuru Profiler by an agent is used to create application profiles for visualizations:

- codeguru-profiler:GetProfile
- codeguru-profiler:DescribeProfilingGroup

For more information, see Working with visualizations.

The following code is an example of an IAM permission policy statement that grants an identity access to retrieve profiler data for a specific profiling group. Limiting access to individual resources provides a more secure approach when allowing this access, as it ensures you grant only the permission you intend without risking accidental access to additional resources.

```
{"Statement": [{"Effect": "Allow",
      "Action": [
         "codeguru-profiler:GetProfile",
         "codeguru-profiler:DescribeProfilingGroup"
      ],
      "Resource": "arn:aws:codeguru-
profiler:<region>:<accountID>:profilingGroup/profilingGroupName"
   }]
```

```
}
```

For users accessing this data in the CodeGuru Profiler console, you also need to have an additional policy statement providing ListProfilingGroups permissions to allow users to see the list of the profiling groups in the account. For example, the following code allows users to see a list of all profiling groups in their AWS account and Region.

```
{"Statement": [{"Effect": "Allow",
      "Action": [
         "codeguru-profiler:ListProfilingGroups"
      ],
      "Resource": "*"
   }]
}
```

For more information, see ConfigureAgent, PostAgentProfile, GetProfile, DescribeProfilingGroup, and ListProfilingGroups in the Amazon CodeGuru Profiler API Reference.

The minimum required permissions to be applied to a role used for collecting data using the profiling agent, scoped to a single profiling group is:

```
{"Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "codeguru-profiler:ConfigureAgent",
                "codeguru-profiler:PostAgentProfile"
            ],
            "Resource": "arn:aws:codeguru-
profiler:<region>:<accountID>:profilingGroup/<profilingGroupName>"
        }
    ]
}
```

A full list of the available actions, resources and condition keys which can be used to write further least privilege policies specific to your environment is available in the Service Authorization Documentation.

Using tags to control access to Amazon CodeGuru Profiler resource

Conditions in IAM policy statements are part of the syntax that you can use to specify permissions for CodeGuru Profiler profiling group-based actions. You can create a policy that allows or denies actions for profiling groups based on the tags associated with those profiling groups, then apply those policies to the IAM groups configured for managing IAM users. For information about applying tags to a profiling group, see Tagging profiling groups.

Example 1: Give all CodeGuru Profiler permissions to the role

The first statement gives all CodeGuru Profiler permissions to all groups with the role. The second statement provides deny permissions to delete any profiling group with tag `{stage: prod}` from the role.

```
{"Version": "2012-10-17",
    "Statement": [
        {"Action": [
                "codeguru-profiler:*"
            ],
            "Effect": "Allow",
            "Resource": "*"
        },
        {"Effect": "Deny",
            "Action": [
                "codeguru-profiler:DeleteProfilingGroup"
            ],
            "Condition": {"StringEquals": {"aws:ResourceTag/stage":
"prod"
                }
            },
            "Resource": "*"
        }
    ]
}
```

Example 2: Deny tagging and untagging a resource

The following policy prevents a role from tagging or untagging a resource if the resource is marked with the tag `{stage: prod}`.

```
{"Version": "2012-10-17",
    "Statement": [
        {"Action": [
                "codeguru-profiler:*"
            ],
            "Effect": "Allow",
            "Resource": "*"
        },
        {"Effect": "Deny",
            "Action": [
                "codeguru-profiler:TagResource",
                "codeguru-profiler:UntagResource"
            ],
```

```
            "Condition": {"StringEquals": {"aws:ResourceTag/stage":
"prod"
                }
            },
            "Resource": "*"
        }
    ]
}
```

Service-linked roles for CodeGuru Profiler

Amazon CodeGuru Profiler uses AWS Identity and Access Management (IAM) service-linked roles. A service-linked role is a unique type of IAM role that is linked directly to CodeGuru Profiler service and defines the permissions it has to view data outside of the domain of the CodeGuru Profiler service. Service-linked roles are predefined by CodeGuru Profiler and include all the permissions that the service requires to call other AWS services on your behalf.

The role permissions policy allows CodeGuru Profiler to complete the following actions on the specified resources.

```
{"Version": "2012-10-17",
  "Statement": [
    {"Sid": "AllowSNSPublishToSendNotifications",
      "Effect": "Allow",
      "Action": [ "sns:Publish" ],
      "Resource": "*"
    }
  ]
}
```

You must configure permissions to allow an IAM entity (such as a user, group, or role) to create, edit, or delete a service-linked role. For more information, see Service-Linked Role Permissions in the IAM User Guide.

You don't need to manually create a service-linked role. When configuring notifications on any profiling group for the first time, you configure an Amazon Simple Notification Service (Amazon SNS) topic for forwarding notifications from CodeGuru Profiler to subscribers of the Amazon SNS topic. When creating the first notification configuration, CodeGuru Profiler automatically creates the IAM service-linked role, which you can see in the IAM console. This role does not require manual configuration.

By default the service-linked role allows CodeGuru Profiler to send notifications to any SNS topic in your account – per notification settings in a Profiling Group. To maintain least privilege, this policy can be modified for the specific SNS topics your environment uses for CodeGuru Profiler notifications.

Logging

The logging and monitoring of API calls are key components in security and operational best practices, as well as requirements for industry and regulatory compliance. AWS CloudTrail is a web service that records API calls to supported AWS services in your AWS account and delivers a log file to your Amazon S3 bucket.

CodeGuru Profiler is integrated with AWS CloudTrail. By enabling CloudTrail in the AWS Management Console, you can receive a history of CodeGuru API calls made on your account. The Amazon API call history produced by CloudTrail enables security analysis, resource change tracking, and compliance auditing. Calls captured include calls from the CodeGuru console and code calls to the CodeGuru API operations. For more information, see Logging CodeGuru Profiler API calls with AWS CloudTrail.

Monitoring

You can use Amazon CloudWatch to monitor the number of recommendations created for your source code for an associated profiling group. You can track the count of recommendations across a profiling group over time. You can set a CloudWatch alarm that notifies you when the number of recommendations exceeds a threshold you set. For example, you can specify that an Amazon SNS notification is sent when more than 25 recommendations are generated for a branch in a repository within an hour. For more information, see Monitoring CodeGuru Profiler with Amazon CloudWatch.

Conclusion

As an Amazon CodeGuru customer, security is a shared responsibility between AWS and you. CodeGuru is built using best practices on top of AWS technologies, and it provides customers the tools they need to secure their data. Customers can implement best practices to improve their security posture in CodeGuru. This includes configuration of identity, network, logging and monitoring, as well as specific configuration of the CodeGuru service.

Contributors

Contributors to this document include:

- Brian Farnhill, Developer Specialist Solutions Architect, AWS
- James Kingsmill, Solutions Architect, AWS
- Justin Dray, Solutions Architect, AWS

Document Revisions

Date	Description
May 4, 2021	First Publication