# What is secret scanning?

Definition: Secret scanning is a GitHub Advanced Security feature that scans repositories for known types of secrets, such as authentication credentials, to prevent fraudulent use of accidentally committed secrets.

Function: Secret scanning automatically scans the entire Git history of a GitHub repository, across all branches, for any secrets with known patterns.

Notification: When a secret is detected in a private or public repository, secret scanning sends a notification to all repository administrators, informing them about the commit that contains the secret.

Security tab: Repository administrators can access the list of all detected secrets in the Security tab of the repository.

Service provider notification: GitHub notifies the service provider associated with the secret, allowing them to take appropriate actions like revoking the secret or issuing a new one.

Availability: Secret scanning is available for all public repositories by default and can't be turned off. It can be manually enabled for private repositories owned by organizations on an enterprise plan with GitHub Advanced Security.

Custom patterns: If the secrets used in an organization are not automatically detected, custom patterns can be created to ensure they are recognized by secret scanning.

# Configure secret scanning

Enable secret scanning for a private repository:

Navigate to Settings > Security & analysis in the repository.

Click the Enable button next to GitHub Advanced Security.

Review the impact and click Enable GitHub Advanced Security for this repository.

Click the Enable button next to Secret scanning.

Enable secret scanning for an organization:

Go to Settings > Security & analysis in the organization.

Click the Enable all button next to GitHub Advanced Security.

Review the impact and click Enable all.

Click the Enable all button next to Secret scanning.

Optionally enable the feature by default for new repositories in the organization.

Exclude files from being scanned:

Create a .github/secret\_scanning.yml file in the repository.

Use the "paths-ignore" parameter to specify the paths you want to exclude from scanning.

You can use special characters like "\*" to filter paths.

Configure recipients of secret scanning alerts:

GitHub notifies repository administrators, organization owners, and authors of commits that trigger alerts.

Repository administrators and organization owners can give view access to security alerts to others under Settings > Security & analysis.

Type the name of the person or team in the "Search for people or teams" field, select the desired matches, and click Save.

# Use secret scanning

Here are the most important areas regarding the use of secret scanning:

Responding to an alert:

Consider a committed secret compromised and take appropriate actions.

For compromised GitHub personal access tokens, delete the compromised token, create a new token, and update services that use the old token.

For other secrets, verify their validity, create a new secret, update services using the old secret, and delete the old secret.

Resolve the alert by selecting the appropriate reason in the Close as drop-down menu.

Creating a custom pattern for a private repository:

Navigate to Settings > Code security & analysis in the repository.

Under GitHub Advanced Security > Secret scanning, click New pattern.

Provide the name, pattern of the secret (Hyperscan regex), and a sample test string for matching.

Optionally, specify additional match requirements or surrounding content using the More options drop-down and click Create pattern.

Creating a custom pattern for an organization:

Go to Settings > Code security & analysis in the organization.

Under GitHub Advanced Security > Secret scanning, click New pattern.

Provide the name, pattern of the secret (Hyperscan regex), and a sample test string for matching.

Optionally, specify additional match requirements or surrounding content using the More options drop-down and click Create pattern.

Remember to follow best practices for handling compromised secrets and regularly review and update your custom patterns to ensure effective secret scanning.

You can create up to 500 custom patterns for each organization or enterprise account in GitHub Advanced Security. These custom patterns allow you to define additional secret scanning patterns beyond the ones automatically detected by GitHub.

# What are custom patterns?

Custom patterns in the context of secret scanning refer to user-defined patterns that can be created to scan for specific secrets or sensitive information in code repositories. While GitHub automatically scans for known secret patterns, custom patterns allow users to define their own unique patterns to search for specific types of secrets that may be specific to their organization or project.

With custom patterns, users can extend the scanning capabilities of secret scanning to identify additional types of secrets that are not covered by the default patterns. This feature is particularly useful for organizations that have specific requirements or unique secret formats that need to be detected and prevented from being committed to repositories.

Custom patterns are defined using Hyperscan regular expressions, which provide powerful and flexible pattern matching capabilities. Users can specify the name of the pattern, the Hyperscan regex pattern itself, and even provide a sample test string to ensure the configuration matches the expected patterns. Custom patterns can be created at the organization level or for individual private repositories, allowing for fine-grained control over secret scanning customization.

By utilizing custom patterns, organizations can enhance the security of their repositories by proactively detecting and preventing the accidental exposure of sensitive information or secrets.

## Example

Here's an example of a custom pattern using Hyperscan regex for detecting a specific secret format, such as an API key:

Pattern name: API Key Pattern

Hyperscan regex pattern: \b[A-Za-z0-9-\_]{32}\b

In this example, the custom pattern is named "API Key Pattern," and the Hyperscan regex pattern "\b[A-Za-z0-9-\_]{32}\b" is used to match a specific format of an API key. Let's break down the regex pattern:

\b: Represents a word boundary, ensuring that the pattern matches complete words.

[A-Za-z0-9-\_]: Matches any uppercase or lowercase letter, digit, hyphen, or underscore.

{32}: Specifies that the previous character class should occur exactly 32 times, representing a 32-character length API key.

\b: Another word boundary to ensure a complete match.

By defining this custom pattern, secret scanning will search for strings that match the specified format of an API key (32 characters long, consisting of alphanumeric characters, hyphens, or underscores) within the scanned code repositories. If a matching string is found, it will trigger an alert, allowing you to take appropriate action to protect the exposed secret.

Remember to adjust the pattern according to your specific secret format or requirements, and consider any additional context or content that may be relevant when creating custom patterns for secret scanning.

Here are the bullet points summarizing the key points of the module on secret scanning:

Secret scanning helps identify and prevent accidental exposure of secrets in your code repositories.

It is enabled by default on public repositories and can be enabled on private repositories with GitHub Advanced Security.

Secret scanning scans the code on all branches of your repository for known secret patterns.

When a secret is committed, it sends alerts to repository administrators about the affected commit.

Actions to take after an alert depend on the type of secret, such as deleting compromised tokens and creating new ones.

Custom patterns can be created to scan for specific secrets that are not automatically detected by GitHub.

Configuration options allow you to exclude files from scanning, define notification recipients, and create custom patterns.

Secret scanning improves code security and helps prevent unauthorized access to external services.

Regularly reviewing and updating custom patterns is important to ensure effective secret scanning.

Properly responding to secret scanning alerts helps protect your code and sensitive information.

# Security Features

Here is a summary of the section on GitHub's security features:

GitHub offers various security features to protect code and secrets in repositories and organizations. These features include:

* Security policy: Allows users to confidentially report security vulnerabilities found in a repository.
* Security advisories: Enables private discussions and fixes for security vulnerabilities in repository code, with the option to publish advisories to notify the community.
* Dependabot alerts and security updates: Notifies about dependencies with known security vulnerabilities and generates pull requests for automatic updates.
* Dependabot version updates: Automatically raises pull requests to keep dependencies up-to-date, reducing exposure to older versions.
* Dependency graph: Visualizes dependencies and their relationships, helping understand the ecosystem and identify potential vulnerabilities.
* Security overview for repositories: Displays enabled security features and allows configuration of additional ones.
* Secret scanning alerts: Automatically detects leaked secrets and credentials in repositories, notifying users and relevant service providers.
* Code scanning: Automatically detects security vulnerabilities and coding errors in new or modified code before merging.
* Dependency review: Provides insights into the impact of dependency changes and highlights vulnerable versions before merging pull requests.

These features are available for different types of repositories, including public repositories, private repositories with GitHub Advanced Security, and GitHub Enterprise Cloud with a license for GitHub Advanced Security. The GitHub Advisory Database provides a curated list of security vulnerabilities, and the dependency graph can be exported as a Software Bill of Materials.