GitHub Advanced Security implementation guide

WRITTEN BY GITHUB WITH ♥

What's inside

Introduction

GitHub Advanced Security enables you to better secure your supply chain and simplify the remediation process for code in progress. The following guide has been created to help you enable and configure GitHub Advanced Security, and this guide contains direct links to all the resources and documents you might need during your implementation.

Prerequisites:

Your GitHub account manager has notified you that GitHub Advanced Security has been enabled for your enterprise or organization.

You must be the admininstrator or owner of your organization to enable and configure GitHub Advanced Security. If you are not the org admin/owner, you will need to request your GitHub org admin/owner to enable GitHub Advanced Security.

More resources:

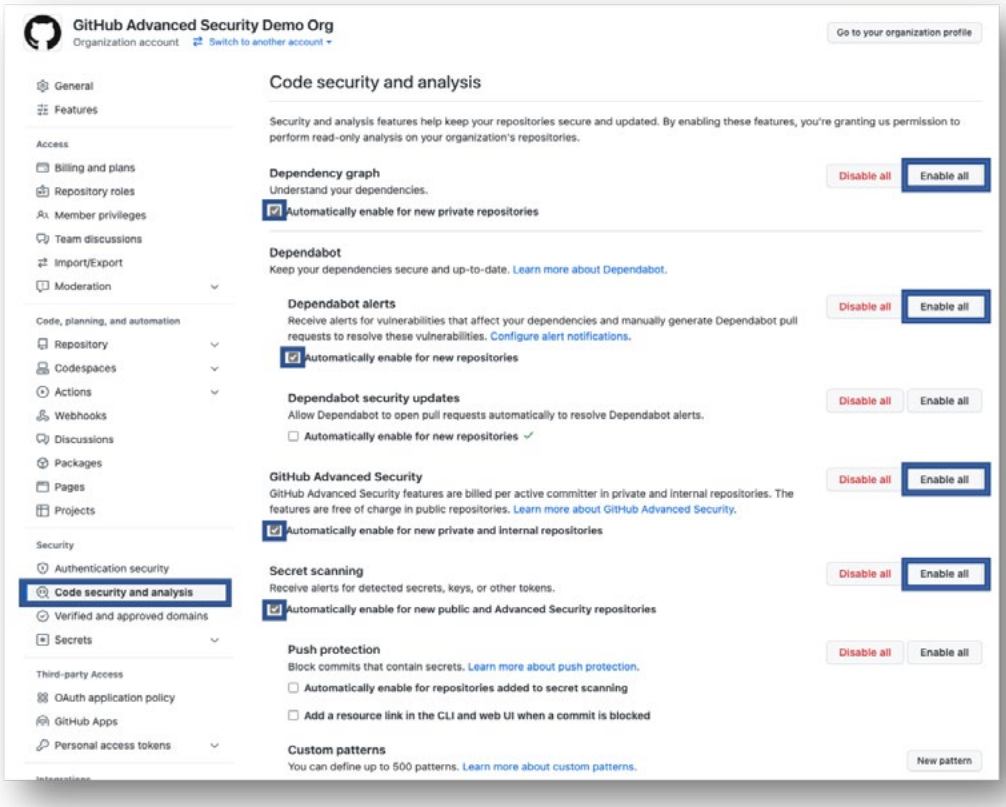
- Code scanning supported languages
- Code scanning third party integration list
- Secret scanning for private repositories
- Push protection for secret scanning
- Code scanning
- Security overview
- Dependency review

Enabling GitHub Advanced Security

GitHub Advanced Security – organizational level

When you enable GitHub Advanced Security at the organizational level, you can secure the code in your organization's private repositories. The following steps address how to configure security features to meet your organization's specific requirements:

1. First access your organization as the administrator or owner, and then click on **Settings**.



2. In **Settings,** click on **Code security and analysis** located under **Security** in the left-side menu panel.

3. Proceed to enable the following:
    - Dependency graph (also enable: **Automatically enable for new private repositories**).
    - Dependabot alerts (also enable: **Automatically enable for new private repositories**).
    - GitHub Advanced Security (also enable: **Automatically enable for new private and internal repositories**).
    - Secret scanning (also enable: **Automatically enable for new public and Advanced Security repositories**).

Please note: It's not recommended that you enable any of the following during the start of your implementation:

4. Dependabot security updates. Enabling this setting will allow Dependabot to open pull requests automatically to resolve Dependabot alerts. This is one of Dependabot's great features, but right now, if Dependabot finds 100 vulnerable dependencies, it will automatically open 100 pull requests. This is not the case if Dependabot alerts are enabled while Dependabot security updates are disabled.
5. Push protection. This feature will block commits that contain secrets. For the start of your implementation, we advise not to enable it as this could disrupt your developers by preventing them from pushing their changes into GitHub if a secret is identified within their push to GitHub.

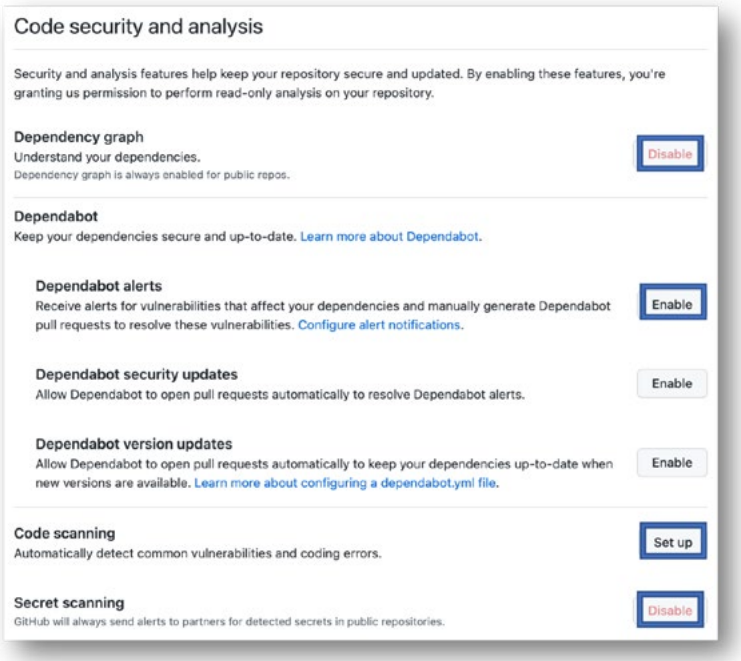
GitHub Advanced Security – repository level

When you enable GitHub Advanced Security at the repository level, you can secure the code in your public repositories. Note that dependency graph and secret

scanning are permanently enabled. The following steps address configuring and managing security features to meet your repo's specific requirements.

1. Within the organization where you have enabled GitHub Advanced Security access any of the repositories you selected for this implementation.
2. Within the repository, click on **Settings** from the navigation bar.
3. Within the repository's **Settings,** click on **Code security and analysis** located under **Security** in the left-side menu panel.
4. Validate that only the following settings are enabled:

    - Dependency graph
    - Dependabot alerts
    - GitHub Advanced Security
    - Code scanning (you will see the option to set it up, which will be covered in this guide)
    - Secret scanning

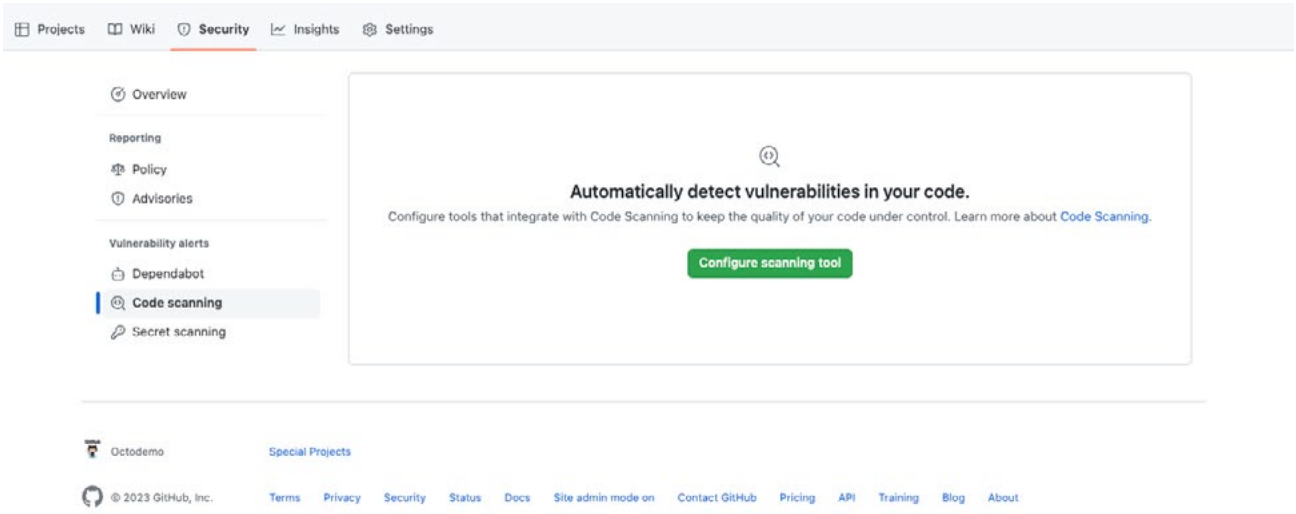


Configuring code scanning

Setting up CodeQL – compiled and interpreted languages

1. Within your repository, proceed on accessing the **Security** tab.
2. In the **Security** tab, access **Code scanning** and click on **Configure scanning tool**.

To help get started with CodeQL, we've created this walkthrough video for reference.



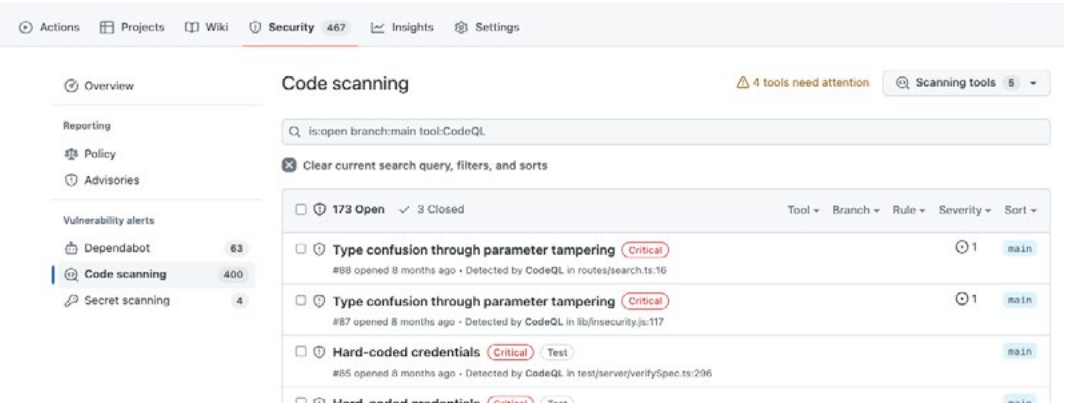
3. Once completed, you will be redirected to edit CodeQL YAML's configuration file. Within that file, you will have the ability to:
    - Change its trigger by changing lines 14 to 21.
    - Add or remove languages on the repository to be scanned by updating line 35.
    - Enable CodeQL's extended security query packs by uncommenting line 53.
    - Customize any setting needed. For details please refer to the following: Configuring code scanning - GitHub Docs

Note: For one of your repositories, we recommend uncommenting line 53 and committing the file to initialize CodeQL on your repository.

4. After reviewing, updating, or changing CodeQL's YAML configuration file, you can proceed with committing the file by clicking on **Start commit** on the top-right corner of your screen.

5. Once you have committed CodeQL's YAML file, proceed to access the **Actions** tab.
6. Within the **Actions** tab, you will see a job being executed. This is CodeQL starting to scan your repository for vulnerabilities. Once completed, it will upload the results. The results are located under the **Security** tab > **Code scanning**. Keep in mind that CodeQL must execute its job before it can upload any results.

Example: The screenshot below is an example of how vulnerabilities will be displayed in code scanning under the **Security** tab if your repo contains any vulnerability that has been identified by CodeQL.



Note: If you do not see any results after configuring CodeQL and allowing it to complete its scan, contact your GitHub account manager so that we can provide you with further assistance during your implementation.

7. In the scenario that you are using a repository where the majority of code is made up of a compiled language, please proceed with configuring code scanning following the steps above. If there is an error or a failure due to CodeQL's auto-builder being unable to build your applilcation, you will need to proceed with the following:

    - Create a bash or PowerShell script file to build your application and upload it to your repository..
    - Once done, you will need to modify CodeQL's YAML file. Comment out lines 58-59 and uncomment lines 67-69. In line 69, you will need to specify the location wihtin the repository of your build script so your application can be successfully built and scanned.
    - After you have completed all the changes needed, proceed with committing the file to initiate CodeQL's execution.

Secret scanning

Understanding secret scanning

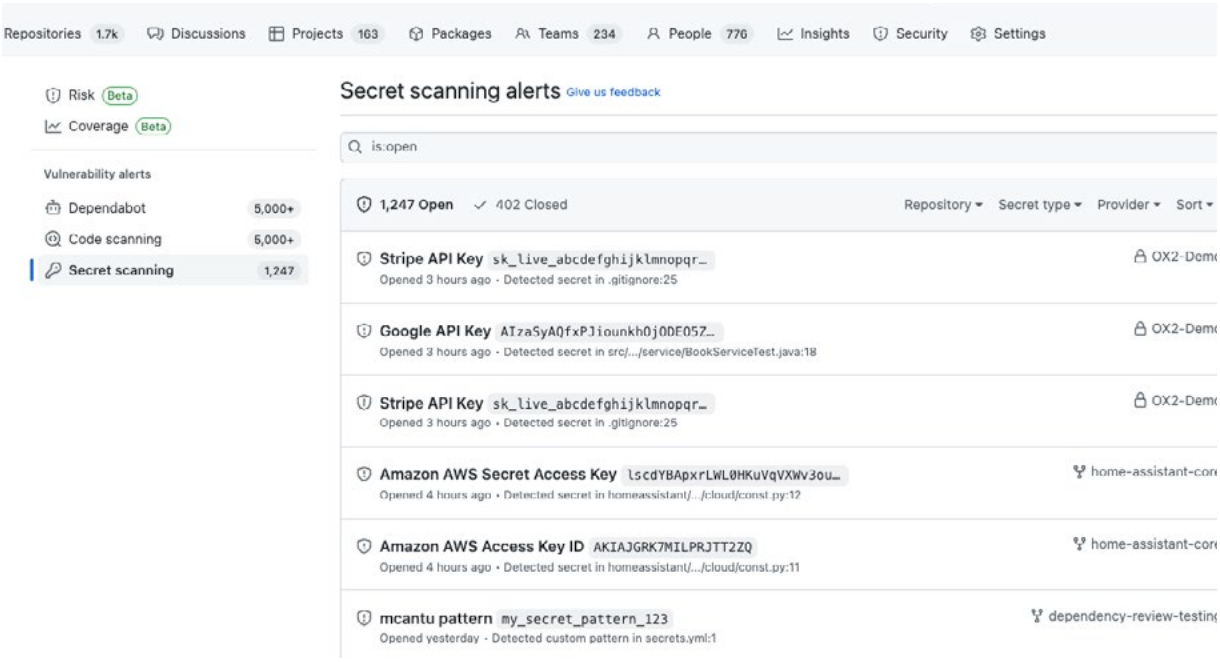
GitHub Advanced Security's secret scanning is a targeted search based on a regular expression. Any strings that match patterns provided by secret scanning partners, by other service providers, or defined by your organization, are reported as alerts in the **Security** tab of repositories. To view the list of our partners to know which secrets we can identify natively, please review the following: Secret scanning patterns – GitHub Docs

In the scenario that your secrets are created in-house or you are generating secrets using a vendor that is not natively supported, you can add support for it by adding a custom pattern so that secret scanning can detect it. Defining custom patterns for secret scanning – GitHub Enterprise Cloud Docs

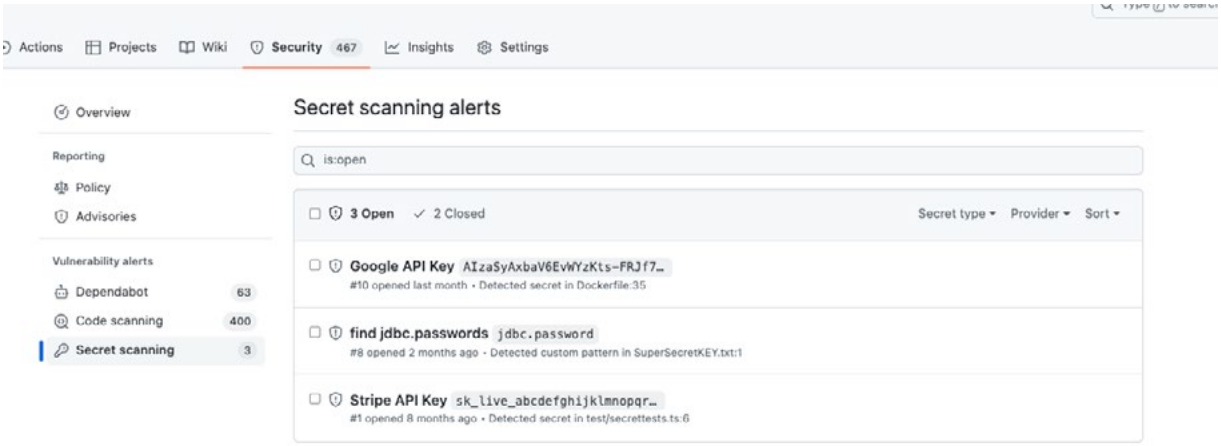
Secret scanning can either be enabled at the organizational level (recommended) or at the repository level. At this point in the guide, you have already enabled secret scanning at the organizational level by following the steps detailed on Page 4. Secret scanning requires no additional configuration, and it will automatically surface any secrets it identifies.

Secrets found can be seen either in specific repositories or at the organizational level.

- **Organizational level:**
  Access the **Organization**> **Security**> **Secret scanning**



- **Repository level:**
  Access any repository within your **Organization**> **Security**> **Secret scanning**



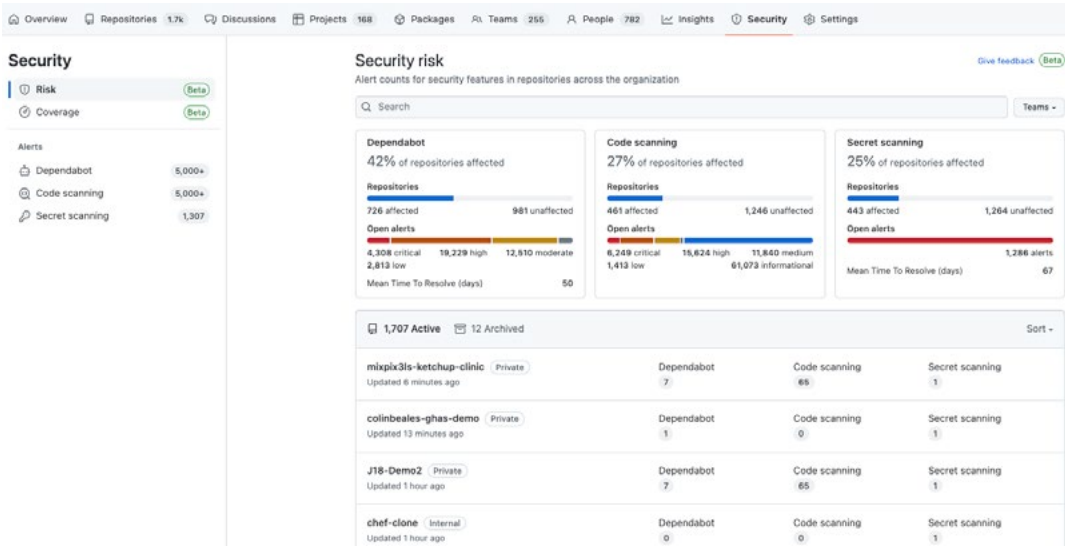
Security overview

Understanding the security overview – organizational level.

Organization owners and security managers can access the security overview for organizations and view their organization's repositories via the enterprise-level security overview. Enterprise owners can use the enterprise-level security overview to view all repositories in their enterprise's organizations. Members of a team can see the security overview for repositories that the team has admin privileges for.

You can use the security overview for a high-level view of the security status of your organization or identify problematic repositories that require intervention. You can view aggregate or repository-specific security information in the security overview. You can also use the security overview to see which security features are enabled for your repositories and to configure any available security features that are not currently in use.

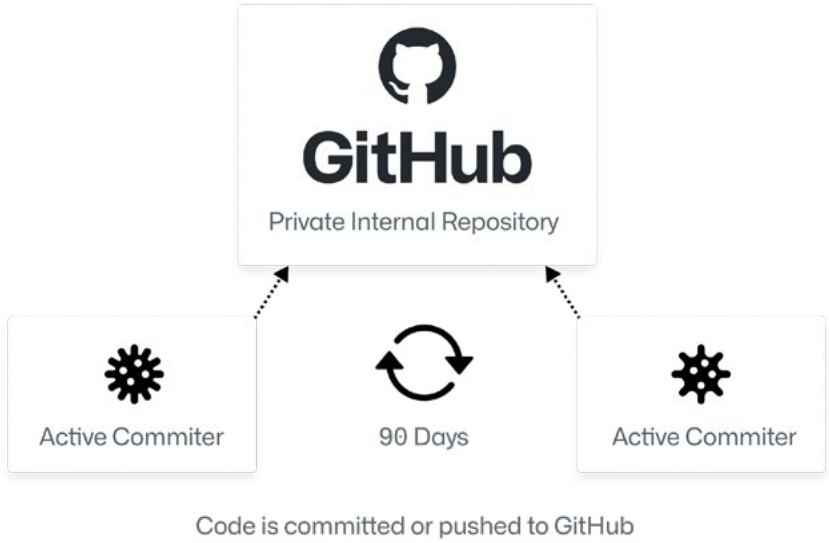
Example: The screenshot below will provide an example of the information you can see from the security overview.



GitHub Advanced Security licensing model

Licenses for GitHub Advanced Security are based on an active committer model. A committer is considered active if one of their commits has been pushed to a private or internal repository within the last 90 days, regardless of when it was originally authored

How does it work?



- **Usage is measured across the whole enterprise account to ensure that each member uses one seat regardless of how many repositories or organizations the user contributes to.**
- **Any user who makes a commit or push to a private or internal repository where GitHub Advanced Security is enabled will consume a license based on a 90-day period.**
- **When you remove a user from your enterprise account, the user's license is freed within 24 hours.**

GitHub Advanced Security Implementation Guide

Learn more at github.com/learn/security
or contact our Sales Team

WRITTEN BY GITHUB WITH ♥