Table of Contents

[Configure Dependabot security updates on your GitHub repo 3](#_Toc132918590)

[Learning objectives 3](#_Toc132918591)

[Prerequisites 3](#_Toc132918592)

[Introduction 4](#_Toc132918593)

[Learning objectives 4](#_Toc132918594)

[Prerequisites 4](#_Toc132918595)

[Manage your dependencies on GitHub 5](#_Toc132918596)

[The dependency graph 5](#_Toc132918597)

[Enable the dependency graph for private repositories 6](#_Toc132918598)

[View the dependency graph 6](#_Toc132918599)

[Supported package ecosystem for the dependency graph 6](#_Toc132918600)

[The GitHub Advisory Database 8](#_Toc132918601)

[Dependabot 8](#_Toc132918602)

[Dependabot alerts 9](#_Toc132918603)

[Set up Dependabot alerts 9](#_Toc132918604)

[Set up Dependabot alerts for private repositories 10](#_Toc132918605)

[Set up Dependabot alerts for organizations 10](#_Toc132918606)

[Set up Dependabot alerts for GitHub Enterprise Server with GitHub Connect 10](#_Toc132918607)

[View Dependabot alerts 10](#_Toc132918608)

[Grant access to Dependabot alerts 11](#_Toc132918609)

[Resolve Dependabot alerts 12](#_Toc132918610)

[The GitHub Advisory Database 13](#_Toc132918611)

[Dependabot 13](#_Toc132918612)

[Dependabot security updates 14](#_Toc132918613)

[Supported repositories 14](#_Toc132918614)

[Manually enable security updates 15](#_Toc132918615)

[View and resolve security updates 15](#_Toc132918616)

[View dependencies being monitored by Dependabot 15](#_Toc132918617)

[Version updates 16](#_Toc132918618)

[Enable version updates for private repositories 17](#_Toc132918619)

[Version updates on forks 18](#_Toc132918620)

[Allow Dependabot to access private repositories 18](#_Toc132918621)

[Security update compatibility scores 18](#_Toc132918622)

[Manage Dependabot notifications and reports 19](#_Toc132918623)

[Configure notifications for Dependabot alerts 20](#_Toc132918624)

[Triage notifications 20](#_Toc132918625)

[View dependency information with the GraphQL API 21](#_Toc132918626)

[GraphQL Explorer 21](#_Toc132918627)

[Write GraphQL queries 22](#_Toc132918628)

[Example API query 22](#_Toc132918629)

[Summary 25](#_Toc132918630)

[Learn more 25](#_Toc132918631)

**Configure Dependabot security updates on your GitHub repo**

Manage your dependencies with GitHub Dependabot

## Learning objectives

By the end of this module, you'll be able to:

* Describe the available tools for managing vulnerable dependencies on GitHub.
* Enable and configure Dependabot alerts.
* Identify the permissions and roles required to view and enable Dependabot alerts.
* Enable and configure Dependabot security updates.
* Identify, review, and address vulnerable dependencies.
* Explain how to use GraphQL API to retrieve vulnerability information.
* Explain how to configure notifications for vulnerable dependencies.

## Prerequisites

* A GitHub account.
* Administrative access to a repository.
* Familiarity with managing GitHub administrative settings.
* Working knowledge of the GitHub pull request workflow.

# Introduction

GitHub dependency management tools help you handle the security risks and maintenance requirements of incorporating third-party software into your project.

Imagine you're responsible for a GitHub project. This project is built on several other pieces of software, also known as dependencies. These dependencies provide necessary functionality but they also require proper management. You want to keep your dependencies up-to-date and put processes in place to address any security risks or vulnerabilities that are introduced because you rely on software that is maintained outside of your project. You also want to automate this process and avoid delays in responding to important issues.

Luckily, GitHub provides you with dependency management tools that help to manage your dependencies and any vulnerabilities they may introduce. In this module, you'll learn about these tools.

## Learning objectives

By the end of this module, you'll be able to:

* Describe the available tools for managing vulnerable dependencies on GitHub.
* Enable and configure Dependabot alerts.
* Identify the permissions and roles required to view and enable Dependabot alerts.
* Enable and configure Dependabot security updates.
* Identify, review, and address vulnerable dependencies.
* Explain how to use GraphQL API to retrieve vulnerability information.
* Explain how to configure notifications for vulnerable dependencies.

## Prerequisites

* A GitHub account.
* Administrative access to a repository.
* Familiarity with managing GitHub administrative settings.
* Working knowledge of the GitHub pull request workflow.

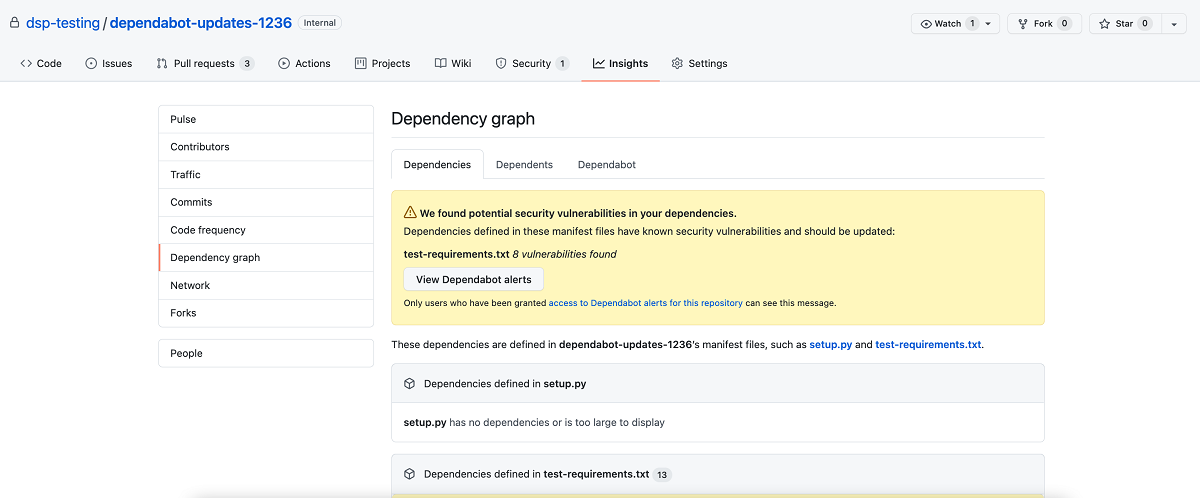
# Manage your dependencies on GitHub

It's common for software projects to depend on external packages or dependencies. Managing these external dependencies can consume resources and affect productivity. These dependencies can also include vulnerabilities that introduce security threats. A vulnerability is a flaw in a project’s code that can be exploited to damage the confidentiality, integrity, or availability of the project or other projects that use its code. You may not even notice vulnerabilities right away because they exist outside of the code that you work on.

In this unit, you'll learn about the GitHub tools for managing your dependencies:

* The dependency graph
* The GitHub Advisory Database
* Dependabot

## The dependency graph



The dependency graph is a summary of a repository's manifest and lock files. These files contain metadata about your project. The dependency graph is automatically generated for public repositories and includes the following information:

* Dependencies, which are the ecosystem and packages the repository depends on.
* Dependents, which are the repositories and packages that depend on the repository.

The dependency graph uses the information from your lock and manifest files to provide a list of three kinds of dependencies:

* The **direct dependencies** explicitly defined in a manifest or lock file.
* The **indirect dependencies**, also known as transitive dependencies or subdependencies, which are dependencies used by packages that are dependencies of your project.
* The **vendored dependencies** that are checked into a specific directory in your repository but aren't referenced in your manifest file (only available for some package managers).

### Enable the dependency graph for private repositories

As a repository administrator, you can also choose to enable the dependency graph for private repositories by completing these steps:

1. Go to your GitHub repository.
2. Select your repository **Settings**.
3. Select **Code security and analysis**.
4. Select **Enable** in the dependency graph section.

Note

Dependent information is not included for private repositories.

### View the dependency graph

You can view the dependency graph for your repository by following these steps:

1. Go to your GitHub repository.
2. Select your repository **Insights**.
3. Select **Dependency graph**.
4. You can select the **Dependencies** or **Dependents** tab from the Dependency graph view.

### Supported package ecosystem for the dependency graph

This table lists the recommended and supported formats for files containing your dependencies. If you use these formats, your dependency graph is more accurate. Using the recommended file format means that your dependency graph reflects the current build set up and can report vulnerabilities for both direct and indirect dependencies.

Lock files are generally recommended in your repository because they define the exact versions of the direct and indirect dependencies that you're currently using. If you're using lock files, make sure the contributors to your repository are also using the same versions. If you use a manifest file (or equivalent), indirect dependencies aren't included in checks for vulnerable dependencies.

|  |  |  |  |
| --- | --- | --- | --- |
| Package manager | Languages | Recommended formats | All supported formats |
| Composer | PHP | composer.lock | composer.json, composer.lock |
| dotnet CLI | .NET languages (C#, C++, F#, VB) | .csproj, .vbproj, .nuspec, .vcxproj, .fsproj | .csproj, .vbproj, .nuspec, .vcxproj, .fsproj, packages.config |
| Maven | Java, Scala | pom.xml | pom.xml |
| npm | JavaScript | package-lock.json | package-lock.json, package.json |
| Python PIP | Python | requirements.txt, pipfile.lock | requirements.txt, pipfile, pipfile.lock, setup.py\* |
| RubyGems | Ruby | Gemfile.lock | Gemfile.lock, Gemfile, \*.gemspec |
| Yarn | JavaScript | yarn.lock | package.json, yarn.lock |

For GitHub Enterprise Security versions 3.2:

|  |  |  |  |
| --- | --- | --- | --- |
| Package manager | Languages | Recommended formats | All supported formats |
| Go modules | Go | go.mod | go.mod |

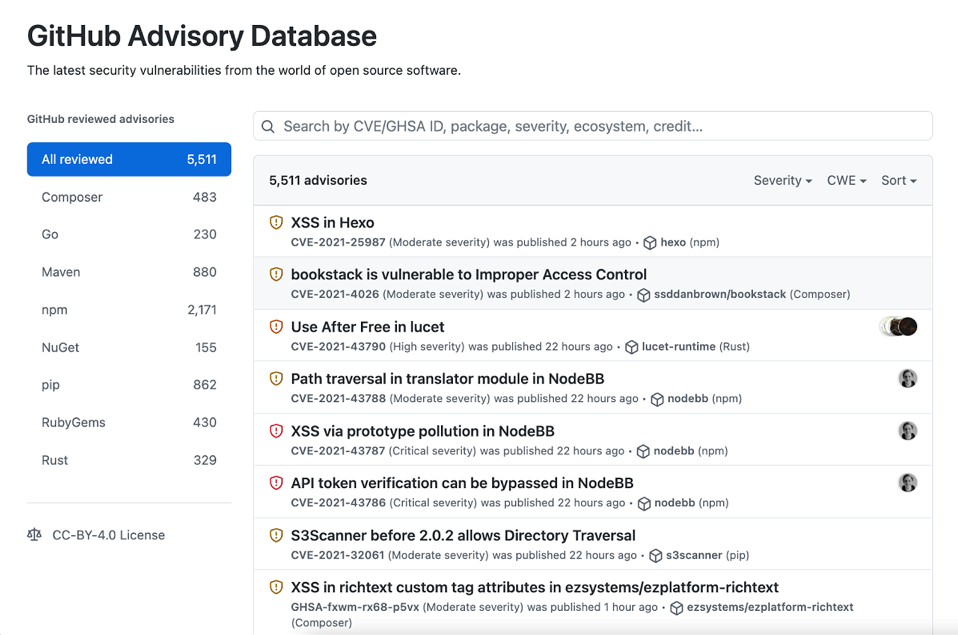
For GitHub Enterprise Security versions above 3.2 and GitHub AE:

|  |  |  |  |
| --- | --- | --- | --- |
| Package manager | Languages | Recommended formats | All supported formats |
| Go modules | Go | go.sum | go.mod, go.sum |

For GitHub Enterprise Security versions above 3.3:

|  |  |  |  |
| --- | --- | --- | --- |
| Package manager | Languages | Recommended formats | All supported formats |
| Python Poetry | Python | poetry.lock | poetry.lock, pyproject.toml |

## The GitHub Advisory Database



GitHub collects information on vulnerabilities and includes it in the GitHub Advisory Database. The GitHub Advisory Database is a curated list of security vulnerabilities related to packages tracked by the GitHub dependency graph. It contains detailed information about each dependency including description, severity, and affected package. This database uses the Common Vulnerability Scoring System (CVSS), Section 5, to assign a severity level to each vulnerability: low, medium/moderate, high, and critical. The database is populated from the following four sources:

* The National Vulnerability Database.
* A combination of machine learning and human reviews to detect vulnerabilities in public commits on GitHub.
* Security advisories reported on GitHub.
* The npm Security advisories database.

## Dependabot

Dependabot is a GitHub tool that automates managing your repository’s dependencies. For Dependabot to work, the dependency graph must be enabled in a repository. Dependabot uses the dependency graph and the GitHub Advisory Database to provide three features:

* **Dependabot alerts**: Notify you about vulnerable dependencies in public repositories.
* **Security updates**: Automatically update or generate a pull request to update vulnerable dependencies.
* **Version updates**: Automatically update all the packages used by your repository.

In the remaining units, you'll learn more about using Dependabot in your repository.

# Dependabot alerts

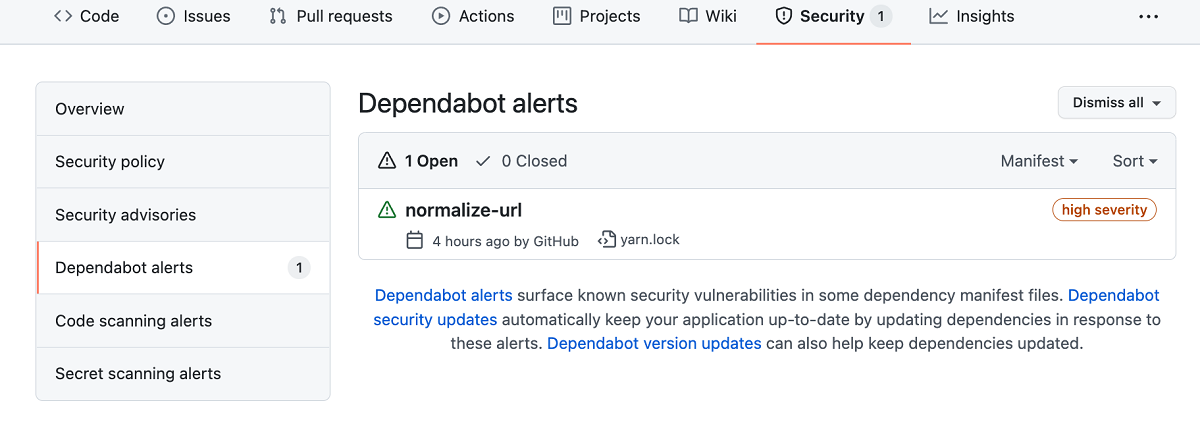
In the previous unit, you learned about the different GitHub tools that work together to enable dependency management and how Dependabot helps to automate some of these processes. Now, you'll learn about Dependabot alerts.

If your project relies on external dependencies, then you can use valuable resources trying to monitor them. This monitoring process is important because you have to be aware of any changes or vulnerabilities in the code. It can also be challenging because this code isn't a part of your project. GitHub helps to automate this process by monitoring your dependencies and then sending Dependabot alerts when vulnerabilities are detected in your repository.

Dependabot alerts are generated under two conditions:

* A new vulnerability is added to the GitHub Advisory Database.
* The dependency graph for a repository changes.

GitHub also reviews pull request attempts to merge changes into the main branch that contain dependency changes. A Dependabot alert is generated if this change would introduce a vulnerability.



Note: GitHub's security features do not claim to catch all vulnerabilities. Though we are always trying to update our vulnerability database and generate alerts with our most up-to-date information, we will not be able to catch everything or tell you about known vulnerabilities within a guaranteed time frame. These features are not substitutes for human review of each dependency for potential vulnerabilities or any other issues, and we recommend consulting with a security service or conducting a thorough vulnerability review when necessary.

## Set up Dependabot alerts

Dependabot alerts are enabled for public repositories by default. Repository administrators and owners can however, set up Dependabot alerts for private repositories and for some GitHub Enterprise Server repositories. Enabling these features grants GitHub permission to perform read-only analysis of those specific repositories.

### Set up Dependabot alerts for private repositories

To set up Dependabot alerts for private repositories, you need to enable both the dependency graph and Dependabot alerts. Follow the steps below for each feature:

1. Sign in to your GitHub account and select your profile photo from the upper right.
2. Select **Settings**, then select **Code security and analysis** under **Security** in the left-side menu.
3. Select **Enable all** to the right of the feature you want to enable.
4. If you would like these settings to be applied to all new repositories in your organization, then select the **Enable by default for new private repositories** checkbox.
5. Select **Enable FEATURE** to enable the feature for all the repositories you own.

### Set up Dependabot alerts for organizations

If you are an organization owner, then you can enable the dependency graph and Dependabot alerts for all repositories in your organization at once:

1. Sign in to your GitHub account and select your profile photo from the upper-right.
2. Select **Your organizations.**
3. Select **Settings** next to the organization for which you would like to enable Dependabot alerts.
4. Select **Code security and analysis** from the left sidebar.
5. On the **Configure security and analysis features** page, select **Enable all** next to the feature you want to turn on.
6. If you would like these settings to be applied to all new repositories in your organization, select the **Enable by default for new private repositories** checkbox.
7. Select **Enable FEATURE** to enable the feature for all the repositories in your organization.

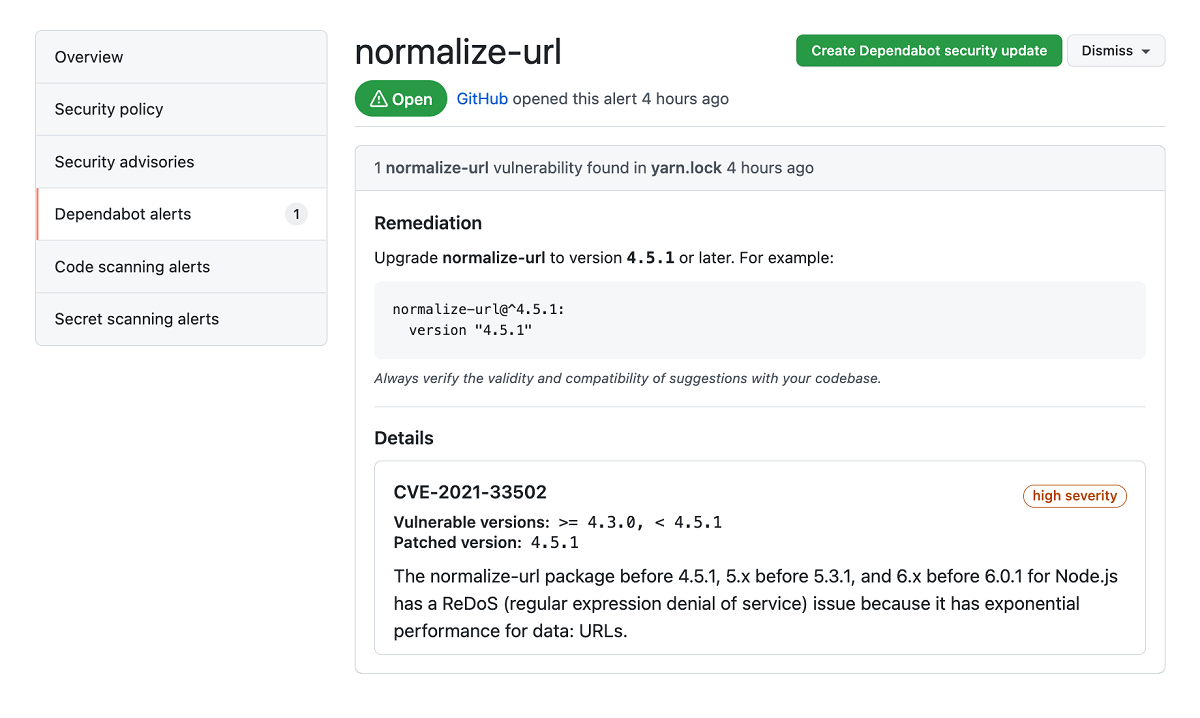
### Set up Dependabot alerts for GitHub Enterprise Server with GitHub Connect

GitHub Enterprise customers can also enable Dependabot alerts. If you're a GitHub Enterprise owner who is also an owner of the connected GitHub Cloud organization or enterprise account, you can use GitHub Connect to enable the dependency graph and Dependabot alerts for your GitHub Server instance. GitHub Connect lets you share certain features and data between your GitHub Enterprise Server instance and your GitHub Enterprise Cloud organization or enterprise account on GitHub.com.

For more information, see [Connect your enterprise account to GitHub Enterprise Cloud](https://docs.github.com/enterprise-server@3.1/admin/configuration/managing-connections-between-your-enterprise-accounts/connecting-your-enterprise-account-to-github-enterprise-cloud).

## View Dependabot alerts

Dependabot alerts are displayed in the **Security** tab for the repository and in the repository's dependency graph. The alert includes a link to the affected file in the project, and information about a fixed version, if available.



## Grant access to Dependabot alerts

By default, only repository owners and administrators are able to receive and dismiss Dependabot alerts for their repositories. Administrators and owners can also grant other teams and users with access to the repository, permissions to view and dismiss Dependabot alerts by following these steps:

1. Go to the main page of the repository.
2. In the left menu, select **Code security and analysis**.
3. In the Access to alerts section, type the name of the person or team that you would like to be able to manage Dependabot alerts in the search bar. Make your selection.
4. Select **Save changes**.

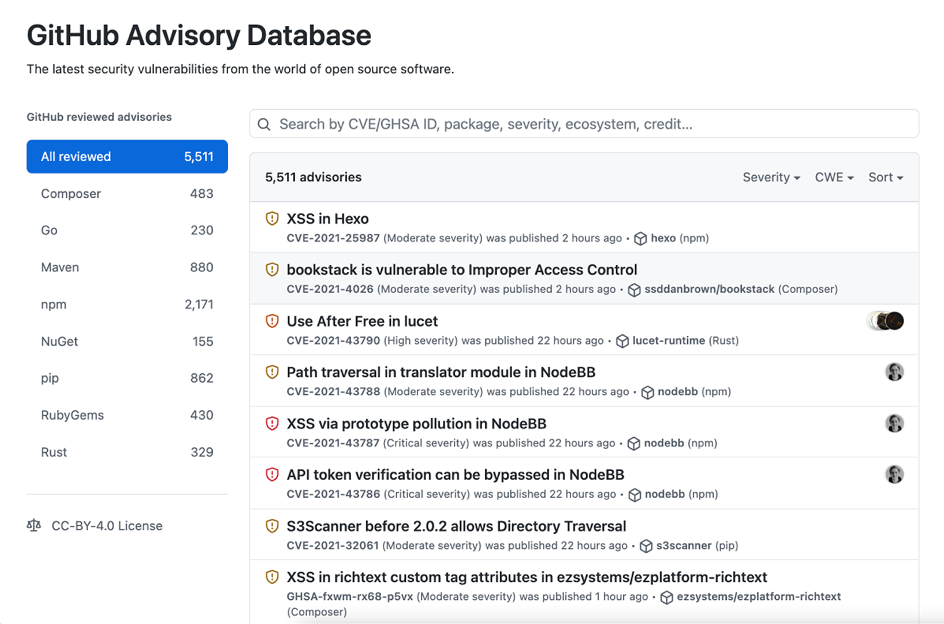
## Resolve Dependabot alerts

After Dependabot alerts are enabled, you should create a process to regularly review and resolve them.

The following steps explain how to resolve Dependabot alerts:

1. Go to the main page of the repository.
2. Select the **Security** tab for the repository.
3. Select **Dependabot alerts** from the security sidebar. A list of the Dependabot alerts for that repository will display.
4. Select the alert you would like to view.
5. Review the alert details. In some cases, the alert may contain a pull request with an automated security update.
6. Resolve the alert by taking one of the following actions:
   * Review and merge the pull request.
   * Select **Create Dependabot security update** to manually fix the vulnerability.
   * Select the **Dismiss** dropdown and choose a reason for dismissing the vulnerability.

## The GitHub Advisory Database



GitHub collects information on vulnerabilities and includes it in the GitHub Advisory Database. The GitHub Advisory Database is a curated list of security vulnerabilities related to packages tracked by the GitHub dependency graph. It contains detailed information about each dependency including description, severity, and affected package. This database uses the Common Vulnerability Scoring System (CVSS), Section 5, to assign a severity level to each vulnerability: low, medium/moderate, high, and critical. The database is populated from the following four sources:

* The National Vulnerability Database.
* A combination of machine learning and human reviews to detect vulnerabilities in public commits on GitHub.
* Security advisories reported on GitHub.
* The npm Security advisories database.

## Dependabot

Dependabot is a GitHub tool that automates managing your repository’s dependencies. For Dependabot to work, the dependency graph must be enabled in a repository. Dependabot uses the dependency graph and the GitHub Advisory Database to provide three features:

* **Dependabot alerts**: Notify you about vulnerable dependencies in public repositories.
* **Security updates**: Automatically update or generate a pull request to update vulnerable dependencies.
* **Version updates**: Automatically update all the packages used by your repository.

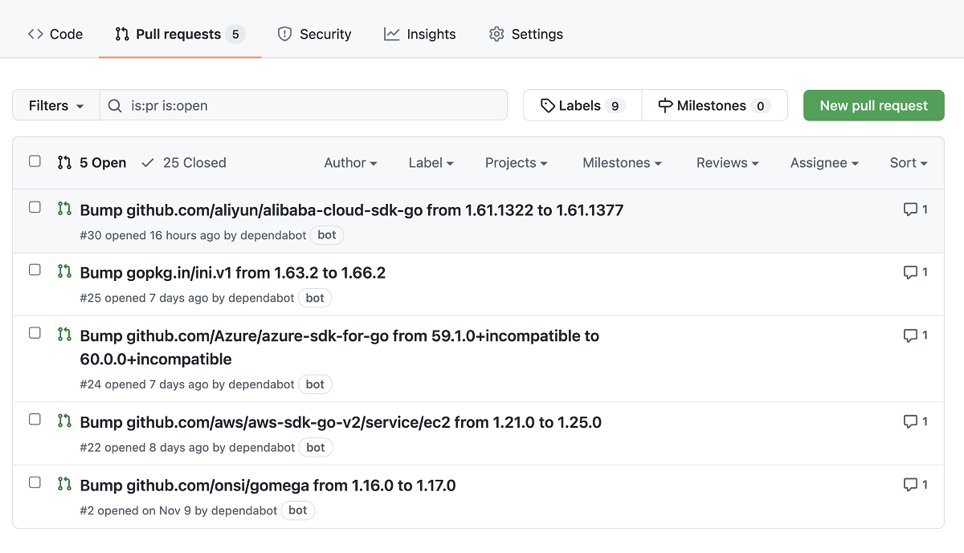
In the remaining units, you'll learn more about using Dependabot in your repository.

# Dependabot security updates

Now you know how Dependabot alerts can help you identify vulnerable dependencies in your repository and you're ready to learn about security updates and version updates. Dependabot security updates help you to fix the vulnerabilities identified by Dependabot alerts. Version updates help manage different versions of dependent packages.

Once you have set up Dependabot alerts to notify you of vulnerabilities in your repository, you can enable two related features so that Dependabot automatically raises pull requests to try to help with your dependency management:

* **Dependabot security updates** are automated pull requests that help you update dependencies with known vulnerabilities.
* **Dependabot version updates** are automated pull requests that keep your dependencies updated, even when they don’t have any vulnerabilities.



## Supported repositories

GitHub automatically enables Dependabot security updates for every repository that meets the following prerequisites:

* The repository isn't a fork.
* The repository isn't archived.
* The repository is public or it's private and you've enabled the read-only analysis by the dependency graph, and vulnerability alerts in the repository's settings.
* Dependabot security updates aren't disabled for the repository.

## Manually enable security updates

You can enable security updates manually on private repositories that don't meet the prerequisites mentioned in the section above. Make sure you've enabled the dependency graph and Dependabot alerts on your repository, and follow these steps to manually enable security updates for it:

1. Sign in to your GitHub account and select your profile photo from the upper-right.
2. Select **Settings** > **Code security and analysis**.
3. Select **Enable** for Dependabot security updates.

## View and resolve security updates

View Dependabot pull requests the same way that you would view other pull requests on GitHub, within the **Pull requests** tab of the related repository. Dependabot pull requests will have dependabot as the author, this account is the bot account used by Dependabot. They'll also use the dependencies label.

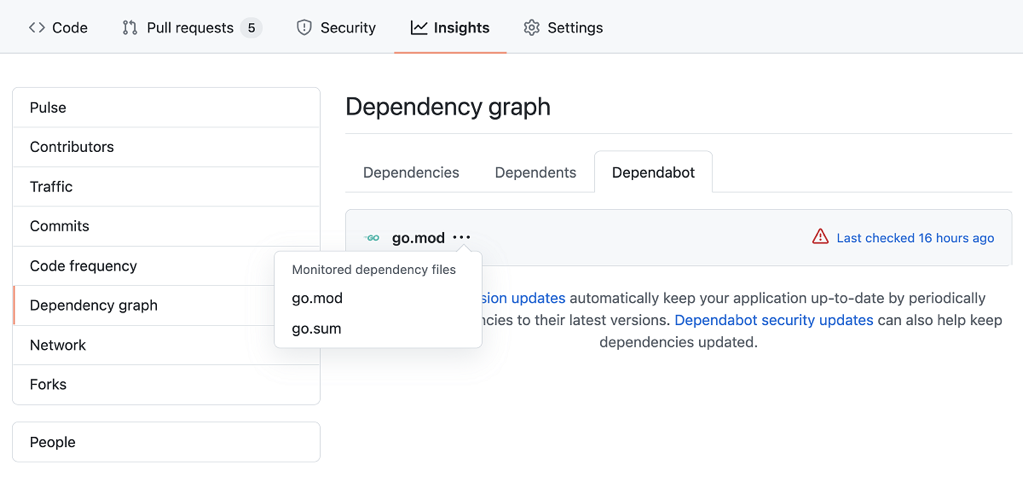
Dependabot pull requests include all the information that you need to review and merge a proposed fix into your project like release notes, changelog entries, and commit details. Details of which vulnerability a pull request resolves are hidden from anyone who doesn't have access to Dependabot alerts for the repository. Merging a security update pull request automatically closes the related Dependabot alert.

Despite all this information, it's still best practice to have automated processes in place so that checks are carried out before the pull request is merged. This approach is especially important if the update includes other functionality or potentially breaking changes.

### View dependencies being monitored by Dependabot

After creating and checking your dependabot.yml file into your repository, you can view the dependencies being monitored by Dependabot to confirm everything is working properly by following the steps below:

1. Go to the main page of the repository.
2. Under your repository name, select **Insights**.
3. Select **Dependency graph** in the left sidebar.
4. Select **Dependabot**.
5. Select the three dots next to a package manager to view the files being monitored.



The **Last checked TIME ago** link in the **Dependabot** tab also enables you to see the log files that Dependabot generated during the last check for version updates. You can rerun the version check by selecting the **Check for updates** button.

## Version updates

Version updates are another Dependabot feature that helps to manage your dependencies by automatically generating a pull request whenever there's a new version of a package or application that your project depends on.

Dependabot checks for new versions by looking at the semantic versioning of the dependency (MAJOR.MINOR.PATCH naming convention), which is included in a manifest or other type of package definition file stored in your repository. To enable version updates, you need to create a dependabot.yml file, which will essentially tell Dependabot where to find the manifest, or other package definition file. You'll learn how to create a dependabot.yml file in the next section.

### Enable version updates for private repositories

Users with write permissions can enable Dependabot version updates for their repository by checking a dependabot.yml file into the .github directory of their repository.

The dependabot.yml file should include the following information:

* version: Should be set to 2.
* registries: Optional if you have dependencies in a private registry. This section contains authentication details.
* updates: Include an entry for each dependency you want Dependabot to monitor.

For each package manager, include:

* package-ecosystem: Specifies the package manager.
* directory: Specifies the location of the manifest or other definition files.
* schedule.interval: Specifies how often to check for new versions.

Below is an example of a dependabot.yml file:

# Basic dependabot.yml file with

# minimum configuration for two package managers

version: 2

updates:

# Enable version updates for npm

- package-ecosystem: "npm"

# Look for `package.json` and `lock` files in the `root` directory

directory: "/"

# Check the npm registry for updates every day (weekdays)

schedule:

interval: "daily"

# Enable version updates for Docker

- package-ecosystem: "docker"

# Look for a `Dockerfile` in the `root` directory

directory: "/"

# Check for updates once a week

schedule:

interval: "weekly"

### Version updates on forks

Dependabot version updates aren't automatically enabled on forks. This safeguard prevents fork owners from unintentionally enabling version updates when they pull changes including a dependabot.yml file from the original repository.

To enable version updates on the fork of a repository:

1. Go to the main page of the repository where you want to enable version updates.
2. Select **Insights**.
3. Select **Dependency graph** from the left sidebar.
4. Select **Dependabot**.
5. Select **Enable Dependabot**.

## Allow Dependabot to access private repositories

To check for outdated dependencies and generate a security update, Dependabot must have access to the repository that contains the dependencies. Dependabot generally can't update dependencies that are located in private repositories or private package registries. However, if the dependency is in a private repository located in the same organization as the package that uses it, then you can grant Dependabot access.

To allow Dependabot to access a private GitHub repository:

1. Go to the security and analysis settings for your organization.
2. Under **Grant Dependabot access to private repository**, select **Add private repositories** or **Add internal and private repositories**.
3. Start typing the name of the repository you want to allow.
4. Select the repository you want to allow.
5. Optionally, to remove a repository from the list, go to the right of the repository and select **X**.

## Security update compatibility scores

Dependabot security updates may include compatibility scores to let you know whether updating a dependency could cause breaking changes. This score is calculated from continuous integration (CI) tests in other public repositories where the same security update has been generated. The compatibility score is the percentage of CI runs that passed when updating between specific versions of the dependency.

# Manage Dependabot notifications and reports

In this unit, you'll learn how to manage your Dependabot notifications. Properly managing these notifications will help you better secure your repository and make sure that important issues are promptly noticed by the right people.

You can always view information about vulnerable dependencies in your repository's **Security** tab. GitHub also notifies repository owners and users with admin permissions by default anytime a new alert is detected. To receive a Dependabot alert, admins must be watching the repository, have enabled notifications for security alerts or all activity on the repository, and must not be ignoring the repository. The Dependabot alerts unit explained how admins can also grant other users access to view Dependabot alerts.

GitHub never publicly shares vulnerability information for any repository. This information is only available to repository owners, people with admin permissions, and users who have been granted the appropriate access as explained in the Dependabot alerts unit.

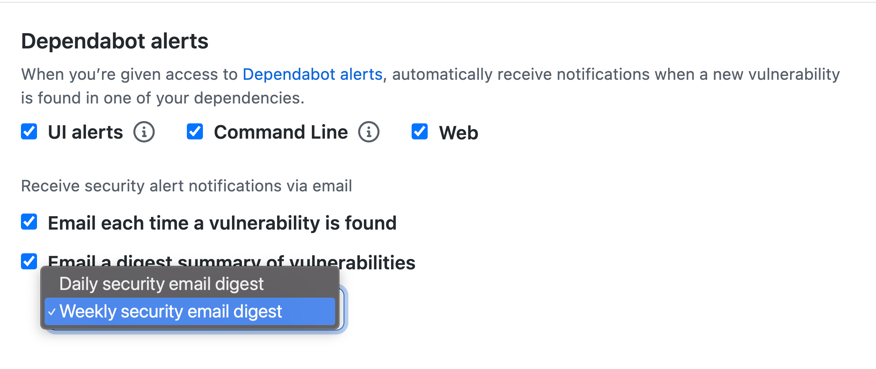
By default, users receive notifications in the following manner:

* By email, an email is sent when Dependabot is enabled for a repository, when a new manifest file is committed to the repository, and when a new vulnerability with a critical or high severity is found.
* In the user interface, a warning is shown in your repository's file and code views if there are any vulnerable dependencies.
* On the command line, warnings are displayed as callbacks when you push to repositories with any vulnerable dependencies.
* In your inbox, as web notifications. A web notification is sent when Dependabot is enabled for a repository, when a new manifest file is committed to the repository, and when a new vulnerability with a critical or high severity is found.
* On GitHub for mobile, as web notifications.

Note: You can sync your web and mobile notifications so that taking action on a notification in one inbox will automatically change its status in the other. For example reading a notification on mobile would change its status to 'read' for the web inbox as well. To sync your web and mobile inboxes, make sure that your email client can view images from notifications@github.com.

## Configure notifications for Dependabot alerts

You can also further customize these notifications by selecting your profile picture and then selecting **Settings** > **Notifications**. In the **Dependabot alerts** section, you can choose how you would like to be notified whenever a new vulnerability is found for a repository that you're watching. You can also choose to subscribe to the weekly or daily security email, which provides a summary of alerts for up to 10 of your repositories.



The security email digest is also a good way to reduce noise from notifications about vulnerable dependencies. If you're concerned about receiving too many notifications, then you can turn off notifications from the **Dependabot alerts** section and instead choose to receive the weekly security digest. You can still view your Dependabot alerts in your repository's **Security** tab. Be sure to include a valid email in the **Email notification preferences** section.

## Triage notifications

You can use the notifications inbox on GitHub and GitHub for mobile to triage your notifications. To use the inbox, you must enable web and mobile notifications from your notifications settings: Select the bell in the upper-right corner of any screen, select **Manage notifications** > **Notification settings**, and then select the **Web and Mobile** checkbox from the **Watching** section.

The following options are available for triaging your security alerts notifications:

* Triage multiple notifications at once.
* Mark completed notifications as 'Done' and remove them from your inbox. To view all of your notifications marked as 'Done', use the is:done query.
* Save a notification to review later. Saved notifications are flagged in your inbox and kept indefinitely. To view all of your saved notifications, use the is:saved query.
* Unsubscribe and remove a notification from your inbox.
* Preview the issue, pull request, or team discussion where the notification originates from on GitHub.com from within the notifications inbox.
* See one of the latest reasons you're receiving a notification from your inbox with a reasons label.
* Create custom filters to focus on different notifications when you want.
* Group notifications in your inbox by repository or date to get a quick overview with less context switching.

## View dependency information with the GraphQL API

Finally, you can also use the GraphQL API to retrieve and export Dependabot alert information. The GraphQL API enables you to export a list of your dependencies that can be saved as a CSV and then imported into a Business Intelligence tool for analysis. You can use the API to quickly audit the known vulnerabilities of dependencies across your organizations.

You'll need to create an OAuth token with the following scopes to communicate with the GraphQL server:

* user
* public\_repo
* repo
* repo\_deployment
* repo:status
* read:repo\_hook
* read:org
* read:public\_key
* read:gpg\_key

The GraphQL API has a single endpoint that doesn't change:

https://api.github.com/graphql

### GraphQL Explorer

The GraphQL Explorer is the recommended tool for communicating with the GraphQL API. The GraphQL Explorer is a "graphical interactive in-browser GraphQL IDE". The Explorer is an instance of GraphiQL. You can download and install the GraphiQL app from <https://github.com/skevy/graphiql-app>.

After downloading the GraphiQL app, follow these instructions to configure it:

1. Get a properly configured OAuth token.
2. Open GraphiQL.
3. In the upper-right corner of GraphiQL, select **Edit HTTP Headers**.
4. In the **Key** field, enter 'Authorization'. In the **Value** field, enter Bearer <token>, where <token> is your generated OAuth token.
5. Select the checkmark to the right of the token to save it.
6. To return to the editor, select outside of the **Edit HTTP Headers** modal.
7. In the **GraphQL Endpoint** field, enter https://api.github.com/graphql.
8. In the **Method** dropdown, select **Post**.

### Write GraphQL queries

GraphQL queries return only the data you specify. When forming a query, you must specify fields within fields (also known as nested subfields) until you return only scalars (primitive values like integers, float, string, boolean, or GitHub object IDs).

Queries are structured like this:

GraphQL

query {

JSON objects to return

}

### Example API query

The following code block is an example of a complex API query that fetches the vulnerable dependency for a repository.

query {

repository(name: "${repo}", owner: "${org}") {

vulnerabilityAlerts(first: 100) {

nodes {

createdAt

dismissedAt

securityVulnerability {

package {

name

}

severity

vulnerableVersionRange

advisory {

ghsaId

publishedAt

identifiers {

type

value

}

}

}

}

}

}

}

The parts of this query are explained below:

query

Your goal is to read data from the server, not to modify it, query is the root operation. (If you don't specify an operation, query is also the default).

repository (name: "${repo}", owner: "${org}") {

Begin the query by finding a repository object. The schema validation indicates this object requires an owner and a name argument.

vulnerabilityAlerts(first: 100) {

The vulnerabilityAlerts object accounts for all Dependabot alerts in the repository. Schema validation indicates this object requires a last or first number of results as an argument, this example uses 100.

nodes {

Retrieve the nodes at the end of the edge.

createdAt

dismissedAt

securityVulnerability {

package {

name

}

severity

vulnerableVersionRange

advisory {

For the nodes, specify the objects to return, in this case createdAt, dismissedAt and securityVulnerability.

GraphQL

ghsaId

publishedAt

identifiers {

type

value

}

}

}

}

}

Specify the ghsaId, publishedAt, and identifiers fields of the advisory object. The identifiers field has the type type and value.

Information about resolved alerts isn't stored in the API and can't be retrieved.

You can also create GitHub Actions that automate the process of retrieving dependency information based on some event or at some interval.

# Summary

The goal of this module was to explain how you can use GitHub Dependabot to help manage your dependencies and identify vulnerabilities that can affect the security of your project.

GitHub Dependabot is enabled by default for all public repositories. You can also use the Dependabot features in private repositories by enabling the dependency graph and Dependabot. When enabled, Dependabot alerts notify you of vulnerabilities in your dependencies and security updates will automatically generate pull requests that try to fix them. Version updates will also automatically generate pull requests to update your dependencies to the latest non-breaking version.

Finally, you learned how to configure your notifications based on how you want to receive alerts about vulnerabilities in your repository. You also learned how to use the security digest email as a concise way to receive a daily or weekly list of vulnerabilities in your repository.

### Learn more

Use these links to learn more about the information covered in this module:

* [GitHub Advisory Database](https://github.com/advisories)
* [About the Dependency graph](https://docs.github.com/en/code-security/supply-chain-security/understanding-your-software-supply-chain/about-the-dependency-graph)
* [Managing security and analysis settings for your repository](https://docs.github.com/en/repositories/managing-your-repositorys-settings-and-features/enabling-features-for-your-repository/managing-security-and-analysis-settings-for-your-repository#granting-access-to-security-alerts)
* [Configuring notifications for vulnerable dependencies](https://docs.github.com/en/code-security/supply-chain-security/managing-vulnerabilities-in-your-projects-dependencies/configuring-notifications-for-vulnerable-dependencies)
* [Enabling the dependency graph and Dependabot alerts on your enterprise account](https://docs.github.com/enterprise-server@3.1/admin/configuration/managing-connections-between-your-enterprise-accounts/enabling-the-dependency-graph-and-dependabot-alerts-on-your-enterprise-account)
* [Connecting your enterprise account to GitHub Enterprise Cloud](https://docs.github.com/enterprise-server@3.1/admin/configuration/managing-connections-between-your-enterprise-accounts/connecting-your-enterprise-account-to-github-enterprise-cloud)
* [Configuring an outbound web proxy server](https://docs.github.com/en/enterprise-server@3.0/admin/configuration/configuring-network-settings/configuring-an-outbound-web-proxy-server)
* [About Dependabot security updates](https://docs.github.com/en/code-security/supply-chain-security/managing-vulnerabilities-in-your-projects-dependencies/about-dependabot-security-updates)
* [Configuring Dependabot security updates](https://docs.github.com/en/code-security/supply-chain-security/managing-vulnerabilities-in-your-projects-dependencies/configuring-dependabot-security-updates)
* [Managing security and analysis settings for your organization](https://docs.github.com/en/organizations/keeping-your-organization-secure/managing-security-and-analysis-settings-for-your-organization)
* [GitHub GraphQL API](https://docs.github.com/en/graphql)