

# Configuring dependency review

## In this article

About dependency review

About configuring dependency review

About configuring the dependency review action

Configuring the dependency review action

You can use dependency review to catch vulnerabilities before they are added to your project.

## About dependency review

Dependency review helps you understand dependency changes and the security impact of these changes at every pull request. It provides an easily understandable visualization of dependency changes with a rich diff on the "Files Changed" tab of a pull request.

Dependency review informs you of:


- Which dependencies were added, removed, or updated, along with the release dates.
- How many projects use these components.
- Vulnerability data for these dependencies.

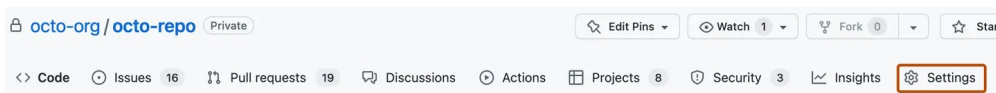
For more information, see "[About dependency review](#)" and "[Reviewing dependency changes in a pull request](#)."


## About configuring dependency review

Dependency review is available when dependency graph is enabled for your GitHub Enterprise Server instance and Advanced Security is enabled for the organization or repository. For more information, see "[Enabling GitHub Advanced Security for your enterprise](#)."

## Checking if the dependency graph is enabled

- 1 On your GitHub Enterprise Server instance, navigate to the main page of the repository.
- 2 Under your repository name, click  **Settings**. If you cannot see the "Settings" tab, select the ... dropdown menu, then click **Settings**.



- 3 In the "Security" section of the sidebar, click  **Code security and analysis**.
- 4 Under "Configure security and analysis features", check if the dependency graph is

enabled.

- 5 If dependency graph is enabled, click **Enable** next to "GitHub Advanced Security" to enable Advanced Security, including dependency review. The enable button is disabled if your enterprise has no available licenses for Advanced Security.

### Configure security and analysis features

Security and analysis features help keep your repository secure and updated. By enabling these features, you're granting us permission to perform read-only analysis on your repository.

**Dependency graph**

Understand your dependencies.  
Contact your GitHub Enterprise administrators to [enable Dependency Graph](#).

**Dependabot alerts**

Receive alerts of new vulnerabilities that affect your dependencies.  
Contact your GitHub Enterprise administrators to [enable Dependabot alerts](#).

**Dependabot security updates**

Easily upgrade to non-vulnerable dependencies.  
Contact your GitHub Enterprise administrators to [enable Dependabot Security Updates](#).

GitHub Advanced Security

Enable

GitHub Advanced Security features are billed per active committer.

## About configuring the dependency review action

The dependency review action scans your pull requests for dependency changes and raises an error if any new dependencies have known vulnerabilities. The action is supported by an API endpoint that compares the dependencies between two revisions and reports any differences.

For more information about the action and the API endpoint, see the [dependency-review-action](#) documentation, and "[Dependency review](#)" in the API documentation.

The following configuration options are available.

Option	Required	Usage
<code>fail-on-severity</code>	×	Defines the threshold for level of severity ( <code>low</code> , <code>moderate</code> , <code>high</code> , <code>critical</code> ). The action will fail on any pull requests that introduce vulnerabilities of the specified severity level or higher.
<code>fail-on-scopes</code>	×	Contains a list of strings representing the build environments you want to support ( <code>development</code> , <code>runtime</code> , <code>unknown</code> ). The action will fail on pull requests that introduce vulnerabilities in the scopes that match the list.
<code>allow-ghsas</code>	×	Contains a list of GitHub Advisory Database IDs that can be skipped during detection.

You can find the possible values for this parameter in the [GitHub Advisory Database](#).

config-file	×	Specifies a path to a configuration file. The configuration file can be local to the repository or a file located in an external repository.
external-repo-token	×	Specifies a token for fetching the configuration file, if the file resides in a private external repository. The token must have read access to the repository.

## Configuring the dependency review action

There are two methods of configuring the dependency review action:


- Inlining the configuration options in your workflow file.
- Referencing a configuration file in your workflow file.

Notice that all of the examples use a short version number for the action ( `v3` ) instead of a semver release number (for example, `v3.0.8` ). This ensures that you use the most recent minor version of the action.

## Using inline configuration to set up the dependency review action

- 1 Add a new YAML workflow to your `.github/workflows` folder.

For `runs-on` , the default label is `self-hosted` . You can replace the default label with the label of any of your runners.

YAML 

```
name: 'Dependency Review'
on: [pull_request]

permissions:
  contents: read

jobs:
  dependency-review:
    runs-on: self-hosted
    steps:
      - name: 'Checkout Repository'
        uses: actions/checkout@v4
      - name: Dependency Review
        uses: actions/dependency-review-action@v3
```

- 2 Specify your settings.

This dependency review action example file illustrates how you can use the available configuration options.

YAML 

```

name: 'Dependency Review'
on: [pull_request]

permissions:
  contents: read

jobs:
  dependency-review:
    runs-on: self-hosted
    steps:
      - name: 'Checkout Repository'
        uses: actions/checkout@v4
      - name: Dependency Review
        uses: actions/dependency-review-action@v3
        with:
          # Possible values: "critical", "high", "moderate", "low"
          fail-on-severity: critical

          # ([String]). Skip these GitHub Advisory Database IDs during
          # detection (optional)
          # Possible values: Any valid GitHub Advisory Database ID from
          # https://github.com/advisories
          allow-ghsas: GHSA-abcd-1234-5679, GHSA-efgh-1234-5679

          # ([String]). Block pull requests that introduce vulnerabilities in
          # the scopes that match this list (optional)
          # Possible values: "development", "runtime", "unknown"
          fail-on-scopes: development, runtime

```

## Using a configuration file to set up dependency review action [🔗](#)

- 1 Add a new YAML workflow to your `.github/workflows` folder and use `config-file` to specify that you are using a configuration file.

For `runs-on`, the default label is `self-hosted`. You can replace the default label with the label of any of your runners.

YAML

```

name: 'Dependency Review'
on: [pull_request]

permissions:
  contents: read

jobs:
  dependency-review:
    runs-on: self-hosted
    steps:
      - name: 'Checkout Repository'
        uses: actions/checkout@v4
      - name: Dependency Review
        uses: actions/dependency-review-action@v3
        with:
          # ([String]). Representing a path to a configuration file local to
          # the repository or in an external repository.
          # Possible values: An absolute path to a local file or an external
          # file.
          config-file: './.github/dependency-review-config.yml'
          # Syntax for an external file: OWNER/REPOSITORY/FILENAME@BRANCH
          config-file: 'github/octorepo/dependency-review-config.yml@main'

```

```
# ([Token]) Use if your configuration file resides in a private
external repository.
# Possible values: Any GitHub token with read access to the private
external repository.
external-repo-token: 'ghp_123456789abcde'
```

- 2 Create the configuration file in the path you have specified.

This YAML example file illustrates how you can use the available configuration options.

YAML



```
# Possible values: "critical", "high", "moderate", "low"
fail-on-severity: critical

# ([String]). Skip these GitHub Advisory Database IDs during detection
(optional)
# Possible values: Any valid GitHub Advisory Database ID from
https://github.com/advisories
allow-ghsas:
- GHSA-abcd-1234-5679
- GHSA-efgh-1234-5679

# ([String]). Block pull requests that introduce vulnerabilities in the
scopes that match this list (optional)
# Possible values: "development", "runtime", "unknown"
fail-on-scopes:
- development
- runtime
```

For further details about the configuration options, see [dependency-review-action](#).

## Legal

© 2023 GitHub, Inc. [Terms](#) [Privacy](#) [Status](#) [Pricing](#) [Expert services](#) [Blog](#)