Configuring dependency review

In this article

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

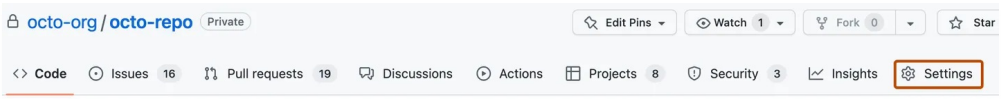
Dependency review is included in GitHub Enterprise Cloud for public repositories. To use dependency review in private repositories owned by organizations, you must have a license for GitHub Advanced Security and have the dependency graph enabled.

Repository administrators can enable or disable the dependency graph for private or internal repositories.

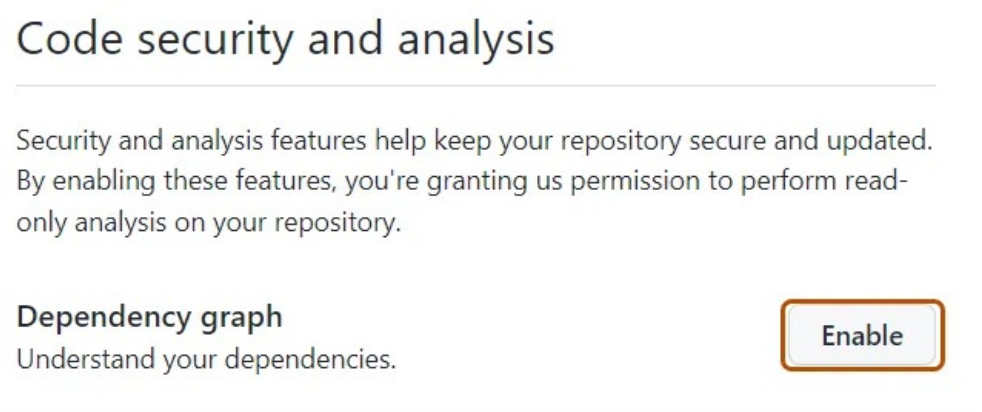
You can enable or disable the dependency graph for all repositories owned by your user account. For more information, see "Managing security and analysis settings for your personal account".

You can also enable the dependency graph for multiple repositories in an organization at the same time. For more information, see "Securing your organization."

1. On GitHub.com, 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. Read the message about granting GitHub Enterprise Cloud read-only access to the repository data to enable the dependency graph, then next to "Dependency Graph", click **Enable**.



   You can disable the dependency graph at any time by clicking **Disable** next to "Dependency Graph" on the settings page for "Code security and analysis."

5. Scroll down the page and if "GitHub Advanced Security" is not enabled, click **Enable** next to the feature.

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
`fail-on-severity`	×	Defines the threshold for level of severity (`low` , `moderate` , `high` , `critical`). The action will fail on any pull requests that introduce vulnerabilities of the specified severity level or higher.
`allow-licenses`	×	Contains a list of allowed licenses. You can find the possible values for this parameter in the Licenses page of the API documentation. The action will fail on pull requests that introduce dependencies with licenses

		that do not match the list.
`deny-licenses`	×	Contains a list of prohibited licenses. You can find the possible values for this parameter in the Licenses page of the API documentation. The action will fail on pull requests that introduce dependencies with licenses that match the list.
`fail-on-scopes`	×	Contains a list of strings representing the build environments you want to support (`development`, `runtime`, `unknown`). The action will fail on pull requests that introduce vulnerabilities in the scopes that match the list.
`allow-ghsas`	×	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.

Tip: The `allow-licenses` and `deny-licenses` options are mutually exclusive.

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.

YAML

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

permissions:
  contents: read

jobs:
  dependency-review:
   runs-on: ubuntu-latest
     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

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

permissions:
  contents: read

jobs:
  dependency-review:
  runs-on: ubuntu-latest
    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


        # You can only include one of these two options: `allow-licenses`
and `deny-licenses`
        # ([String]). Only allow these licenses (optional)
        # Possible values: Any `spdx_id` value(s) from
https://docs.github.com/en/rest/licenses
        allow-licenses: GPL-3.0, BSD-3-Clause, MIT
        # ([String]). Block the pull request on these licenses (optional)
        # Possible values: Any  `spdx_id` value(s) from
https://docs.github.com/en/rest/licenses
        deny-licenses: LGPL-2.0, BSD-2-Clause

        # ([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.

YAML

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

permissions:
 contents: read

jobs:
  dependency-review:
    runs-on: ubuntu-latest
    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

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

  # You can only include one of these two options: `allow-licenses` and
`deny-licenses`
  # ([String]). Only allow these licenses (optional)
  # Possible values: Any `spdx_id` value(s) from
https://docs.github.com/en/rest/licenses
  allow-licenses:
    - GPL-3.0
    - BSD-3-Clause
    - MIT
   # ([String]). Block the pull request on these licenses (optional)
   # Possible values: Any  `spdx_id` value(s) from
https://docs.github.com/en/rest/licenses
  deny-licenses:
    - LGPL-2.0
    - BSD-2-Clause
```

```
   # ([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`.