About code scanning alerts

In this article

Learn about the different types of code scanning alerts and the information that helps you understand the problem each alert highlights.

Code scanning is available for all public repositories on GitHub.com. To use code scanning in a private repository owned by an organization, you must have a license for GitHub Advanced Security. For more information, see "About GitHub Advanced Security."

About alerts from code scanning

You can configure code scanning to check the code in a repository using the default CodeQL analysis, a third-party analysis, or multiple types of analysis. When the analysis is complete, the resulting alerts are displayed alongside each other in the security view of the repository. Results from third-party tools or from custom queries may not include all of the properties that you see for alerts detected by GitHub's default CodeQL analysis. For more information, see "Configuring default setup for code scanning" and "Configuring advanced setup for code scanning."

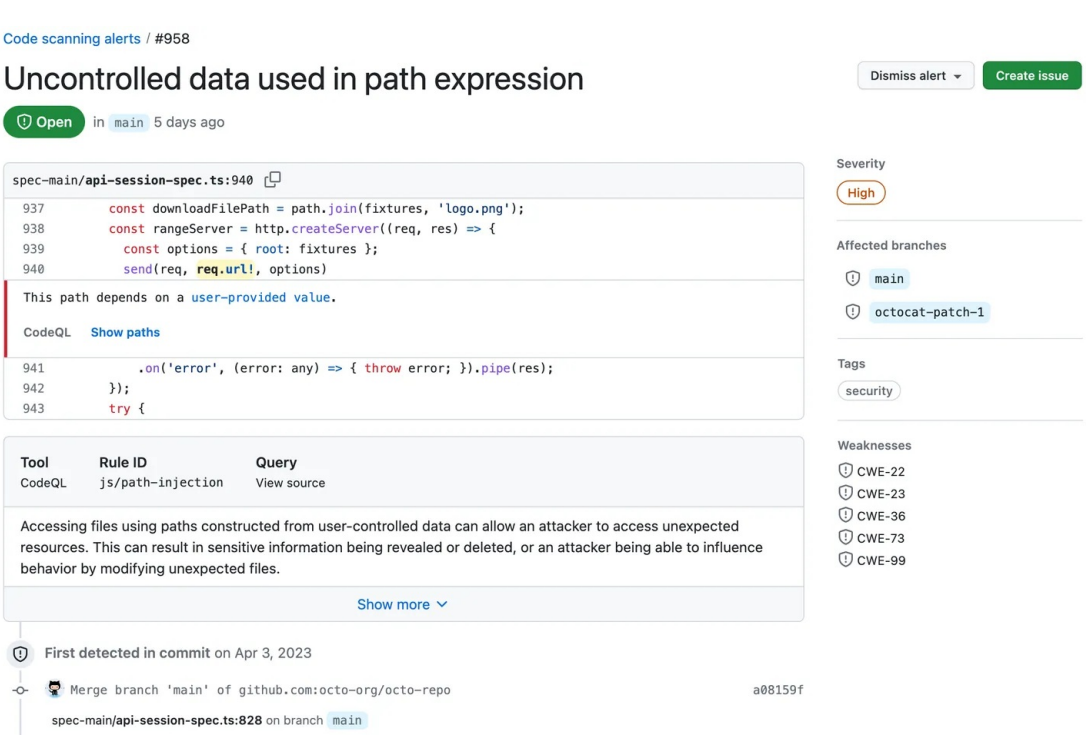
By default, code scanning analyzes your code periodically on the default branch and during pull requests. For information about managing alerts on a pull request, see "Triaging code scanning alerts in pull requests."

You can audit the actions taken in response to code scanning alerts using GitHub tools. For more information, see "Auditing security alerts."

About alert details

Each alert highlights a problem with the code and the name of the tool that identified it. You can see the line of code that triggered the alert, as well as properties of the alert, such as the alert severity, security severity, and the nature of the problem. Alerts also tell you when the issue was first introduced. For alerts identified by CodeQL analysis, you will also see information on how to fix the problem.

The status and details on the alert page only reflect the state of the alert on the default branch of the repository, even if the alert exists in other branches. You can see the status of the alert on non-default branches in the **Affected branches** section on the right-hand side of the alert page. If an alert doesn't exist in the default branch, the status of the alert will display as "in pull request" or "in branch" and will be colored grey.



If you configure code scanning using CodeQL, you can also find data-flow problems in your code. Data-flow analysis finds potential security issues in code, such as: using data insecurely, passing dangerous arguments to functions, and leaking sensitive information.

When code scanning reports data-flow alerts, GitHub shows you how data moves through the code. Code scanning allows you to identify the areas of your code that leak sensitive information, and that could be the entry point for attacks by malicious users.

About severity levels

Alert severity levels may be `Error`, `Warning`, or `Note`.

If code scanning is enabled as a pull request check, the check will fail if it detects any results with a severity of `error`. You can specify which severity level of code scanning alerts causes a check failure. For more information, see "Customizing your advanced setup for code scanning."

About security severity levels

Code scanning displays security severity levels for alerts that are generated by security queries. Security severity levels can be `Critical`, `High`, `Medium`, or `Low`.

To calculate the security severity of an alert, we use Common Vulnerability Scoring System (CVSS) data. CVSS is an open framework for communicating the characteristics and severity of software vulnerabilities, and is commonly used by other security products to score alerts. For more information about how severity levels are calculated, see this blog post.

By default, any code scanning results with a security severity of `Critical` or `High` will cause a check failure. You can specify which security severity level for code scanning results should cause a check failure. For more information, see "Customizing your advanced setup for code scanning."

About alerts from multiple configurations

You can run multiple configurations of code analysis on a repository, using different tools and targeting different languages or areas of the code. Each configuration of code scanning generates a unique set of alerts. For example, an alert generated using the default CodeQL analysis with GitHub Actions comes from a different configuration than

an alert generated externally and uploaded via the code scanning API.

If you use multiple configurations to analyze a file, any problems detected by the same query are reported as alerts generated by multiple configurations. If an alert exists in more than one configuration, the number of configurations appears next to the branch name in the "Affected branches" section on the right-hand side of the alert page. To view the configurations for an alert, in the "Affected branches" section, click a branch. A "Configurations analyzing" modal appears with the names of each configuration generating the alert for that branch. Below each configuration, you can see when that configuration's alert was last updated.

An alert may display different statuses from different configurations. To update the alert statuses, re-run each out-of-date configuration. Alternatively, you can delete stale configurations from a branch to remove outdated alerts. For more information on deleting stale configurations and alerts, see "Managing code scanning alerts for your repository."

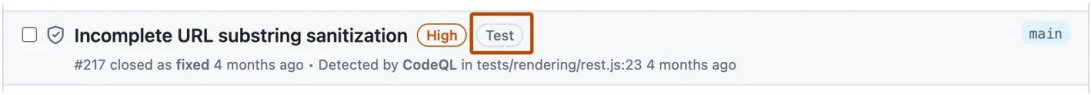
About labels for alerts that are not found in application code

GitHub Enterprise Cloud assigns a category label to alerts that are not found in application code. The label relates to the location of the alert.

- Generated: Code generated by the build process
- Test: Test code
- Library: Library or third-party code
- Documentation: Documentation

Code scanning categorizes files by file path. You cannot manually categorize source files.

In this example, an alert is marked as in "Test" code in the code scanning alert list.

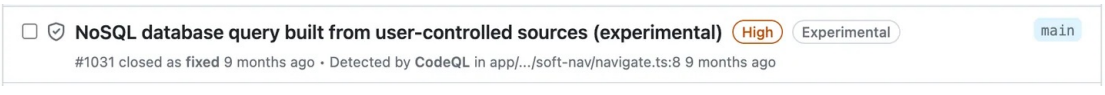


When you click through to see details for the alert, you can see that the file path is marked as "Test" code.

About experimental alerts

Note: Experimental alerts for code scanning are created using experimental technology in the CodeQL action. This feature is currently available as a beta release for JavaScript code and is subject to change.

In repositories that run code scanning using the CodeQL action, you may see some alerts that are marked as experimental. These are alerts that were found using a machine learning model to extend the capabilities of an existing CodeQL query.



Benefits of using machine learning models to extend queries

Queries that use machine learning models are capable of finding vulnerabilities in code that was written using frameworks and libraries that the original query writer did not include.

Each of the security queries for CodeQL identifies code that's vulnerable to a specific type of attack. Security researchers write the queries and include the most common frameworks and libraries. So each existing query finds vulnerable uses of common frameworks and libraries. However, developers use many different frameworks and libraries, and a manually maintained query cannot include them all. Consequently, manually maintained queries do not provide coverage for all frameworks and libraries.

CodeQL uses a machine learning model to extend an existing security query to cover a wider range of frameworks and libraries. The machine learning model is trained to detect problems in code it's never seen before. Queries that use the model will find results for frameworks and libraries that are not described in the original query.

Alerts identified using machine learning

Alerts found using a machine learning model are displayed with an "Experimental alerts" banner to show that the technology is under active development. These alerts have a higher rate of false positive results than the queries they are based on. The machine learning model will improve based on user actions such as marking a poor result as a false positive or fixing a good result.

Enabling experimental alerts

The default CodeQL query suites do not include any queries that use machine learning to generate experimental alerts. To run machine learning queries during code scanning you need to run the additional queries contained in one of the following query suites.

Query suite	Description
`security-extended`	Queries from the default suite, plus lower severity and precision queries
`security-and-quality`	Queries from `security-extended`, plus maintainability and reliability queries

When you update your workflow to run an additional query suite this will increase the analysis time.

```
- uses: github/codeql-action/init@v2
  with:
    # Run extended queries including queries using machine learning
    queries: security-extended
```

For more information, see "Customizing your advanced setup for code scanning."

Disabling experimental alerts

The simplest way to disable queries that use machine learning to generate experimental alerts is to stop running the `security-extended` or `security-and-quality` query suite. In the example above, you would comment out the `queries` line. If you need to continue to run the `security-extended` or `security-and-quality` suite and the machine learning queries are causing problems, then you can contact us through the GitHub Support

portal and open a ticket with the following details.

- Ticket title: "code scanning: removal from experimental alerts beta"
- Specify details of the repositories or organizations that are affected
- Request an escalation to engineering