Code scanning analysis takes too long

In this article

You can fine tune your code scanning configuration to minimize analysis time.

There are several approaches you can try to reduce the build time in a code scanning analysis.

Increase the memory or cores

If you use self-hosted runners to run code scanning analysis, you can increase the memory or the number of cores on those runners. If you're using CodeQL with advanced setup for your analysis, you can review the recommended hardware resources for CodeQL to make sure your self-hosted runners meet those requirements. For more information, see "Recommended hardware resources for running CodeQL."

If you're using GitHub-hosted runners for your code scanning analysis, you could consider upgrading to larger runners. These are GitHub-hosted runners with more RAM, CPU, and disk space than standard runners. For more information about larger runners and the specifications you can use with them, see "About larger runners."

Use matrix builds to parallelize the analysis

To speed up analysis of workflows that involve multiple jobs, you can modify your workflow to use a matrix. For more information, see "Using a matrix for your jobs."

The default CodeQL analysis workflow uses a matrix of languages, which causes the analysis of each language to run in parallel. However, if you're using CodeQL with advanced setup and you have specified the languages you want to analyze directly in the "Initialize CodeQL" step, analysis of each language will happen sequentially. In this configuration, you can speed up your analysis by modifying your advanced setup workflow to use a matrix. For an example, see the workflow extract in "Some languages were not analyzed with CodeQL advanced setup."

Reduce the amount of code being analyzed in a single workflow

Analysis time is typically proportional to the amount of code being analyzed. If you're using CodeQL with advanced setup, you can reduce the analysis time by reducing the amount of code being analyzed at once. For example, by excluding test code, or breaking analysis into multiple workflows that analyze only a subset of your code at a

time.

For compiled languages like Java, Kotlin, Go, C, C++, and C#, CodeQL analyzes all of the code which was built during the workflow run. To limit the amount of code being analyzed, build only the code which you wish to analyze by specifying your own build steps in a `run` block. You can combine specifying your own build steps with using the `paths` or `paths-ignore` filters on the `pull_request` and `push` events to ensure that your workflow only runs when specific code is changed. For more information, see "Workflow syntax for GitHub Actions."

For languages like JavaScript, Python, and TypeScript, that CodeQL analyzes without compiling the source code, you can specify additional configuration options to limit the amount of code to analyze. For more information, see "Customizing your advanced setup for code scanning."

If you split your CodeQL analysis into multiple workflows, we still recommend that you have at least one workflow which runs on a `schedule` which analyzes all of the code in your repository. Because CodeQL analyzes data flows between components, some complex security behaviors may only be detected on a complete build.

Run only during a `schedule` event

You may find that your analysis is slow during `push` or `pull_request` events. If so, you can set your analysis to only trigger on the `schedule` event. If you're using CodeQL for your code scanning analysis, you can configure this with an advanced setup workflow, but not in default setup. For more information, see "Understanding GitHub Actions."

Check which queries or rules the workflow runs

Another option to reduce analysis time is to run only the queries or rules that you consider critical in workflows that run on pull requests. If you use a third-party tool for code scanning, you should refer to the documentation for the tool.

In CodeQL, there are two main query suites available for each language. If you have optimized the CodeQL database build and the process is still too long, you could reduce the number of queries you run. The default query suite is run automatically: it provides the best possible compromise between quality and speed.

If you're using CodeQL with advanced setup, you may be running extra queries or query suites in addition to the default queries. Check whether the workflow defines an additional query suite or additional queries to run using the `queries` element. You can experiment with disabling the additional query suite or queries. For more information, see "Customizing your advanced setup for code scanning."

> **Note:** If you run the `security-extended` or `security-and-quality` query suite for JavaScript, then some queries use experimental technology. For more information, see "About code scanning alerts."