Phase 5: Rollout and scale code scanning

In this article

You can leverage the available APIs to rollout code scanning programmatically by team and by language across your enterprise using the repository data you collected earlier.

This article is part of a series on adopting GitHub Advanced Security at scale. For the previous article in this series, see "Phase 4: Create internal documentation."

Enabling code scanning

Using the data you collated in Phase 2, you can begin to enable GHAS and then code scanning on your repositories, one language at a time. The step-by-step process for enabling GHAS should look like this:

1. Enable GHAS on the repository. For more information, see "Managing security and analysis settings for your repository."
2. Create a pull request against the repository's default branch with a `codeql-analysis.yml` file containing an example of how to run CodeQL for that language. For more information, see "Creating a pull request."
3. Create an issue in the repository to explain why a pull request has been raised. The issue you create can contain a link to the previous communication sent to all users, but can also explain what changes the pull request introduces, what next steps the team have to take, what the team's responsibilities are, and how the team should be using code scanning. For more information, see "Creating an issue."

There is a publicly available tool that completes the first two steps called the ghas-enablement tool. You can re-run the ghas-enablement tool in batches of languages where it makes sense. For example, JavaScript, TypeScript, Python, and Go likely have a similar build process and could therefore use a similar CodeQL analysis file. The ghas-enablement tool can also be used for languages such as Java, C, and C++, but due to the varied nature of how these languages build and compile you may need to create more targeted CodeQL analysis files.

Note: If you are intending to use GitHub Actions to control code scanning and you do not use the ghas-enablement tool, keep in mind that there is no API access to the `.github/workflow` directory. This means that you cannot create a script without a git client underlying the automation. The workaround is to leverage bash scripting on a machine or container which has a git client. The git client can push and pull files into the `.github/workflows` directory where the `codeql-analysis.yml` file is located.

It is important to not just push the `codeql-analysis.yml` file the repository's default branch. Using a pull request puts ownership on the development team to review and merge, allowing the development team to learn about code scanning and involving the team in the process.

You should capture the pull request URLs created by automation, and check each week for any activity and see which ones are closed. After a few weeks, it may be worth creating another issue or sending internal emails if the pull request remains unmerged.

Creating subject matter experts

You can then proceed to the next stage of enablement, which is creating internal subject matter experts (or SMEs) and arranging company meetings. Opening pull requests and issues in repositories will likely tackle a large percentage of your adoption, but this doesn't tackle one-off use cases where a specific build process, framework, or library needs specific feature flags to be enabled. A more personalized and hands-on approach is required to push high adoption, especially for Java, C, and C++.

It's a good idea to run regular company meetings on specific topics to educate and discuss the rollout with a larger group. This is much more time-efficient for an enterprise with thousands of repositories compared to working with one team at a time. Teams can come to sessions that are relevant to them. Some example sessions that have been run before include:

- Code scanning in a container
- Code scanning & Java Struts
- Code scanning & JSP

You can use the data you have collected about the distribution of different languages among repositories to create targeted meetings.

For the next article in this series, see "Phase 6: Rollout and scale secret scanning."