



Security Training with GHAS

Instructor: Andrew Scoppa
email: Andrew.Scoppa@atmosera.com
GH handle: Andrew-Scoppa

Agenda



Setting the scene

Security features and how they fit into a secure development workflow.



Configuring access

Creating teams and applying appropriate permissions.



Reviewing and analyze alerts

Use the integrated reporting facilities to identify common issues and understand risk factors.



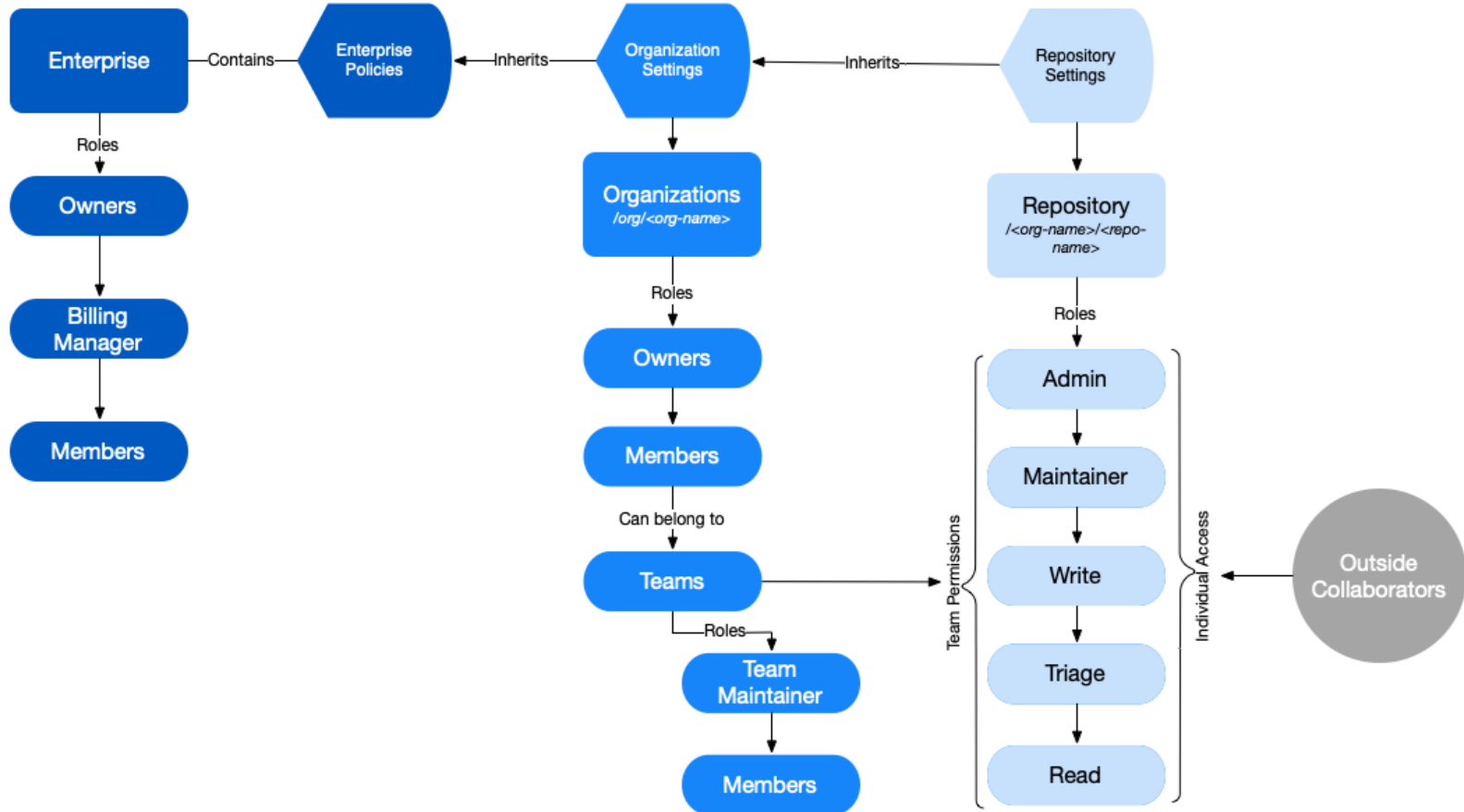
Securing your supply chain

Understanding vulnerabilities in dependencies and patching them.

Resources and Examples

- [Code security documentation](#)
Build security into your GitHub workflow.
- [REST API endpoints for code scanning](#)
Use the REST API to retrieve and update code scanning alerts from a repository.
- [Static Analysis Integrations](#)
A command line application which integrates static analysis tools into the development cycle
- [Removing sensitive data from a repository](#)
Remove unwanted files from a repository's history
- [Advanced Security Material](#)
A place for resources to help you understand and use GitHub Advanced Security (GHAS)
- [Policy as Code](#)
Example application which uses the GHAS APIs to create policy engine using GitHub Actions.
- [GHAS JIRA Integration](#)
A project showing how to integrate GitHub Advanced Security with JIRA.
- [CodeQL Queries](#)
CodeQL queries are used to analyze code for issues related to security, correctness, maintainability, and readability.

Flow of permissions



Repository visibility

- **Public** - Anyone on the internet can access (GHEC only)
- **Internal** - Organization members in the enterprise can access
- **Private** - Only people with explicit access

Create a new repository

A repository contains all project files, including the revision history. Already have a project repository elsewhere? [Import a repository](#).

Repository template

Start your repository with a template repository's contents.

No template ▾

Owner *

droidpl-demorg ▾

Repository name *

Great repository names are short and memorable. Need inspiration? How about [super-duper-memory](#)?

Description (optional)



Public

Anyone on the internet can see this repository. You choose who can commit.



Internal

@droidpl [enterprise members](#) can see this repository. You choose who can commit.



Private

You choose who can see and commit to this repository.

Initialize this repository with:

Skip this step if you're importing an existing repository.



Add a README file

This is where you can write a long description for your project. [Learn more](#).



Add .gitignore

Choose which files not to track from a list of templates. [Learn more](#).

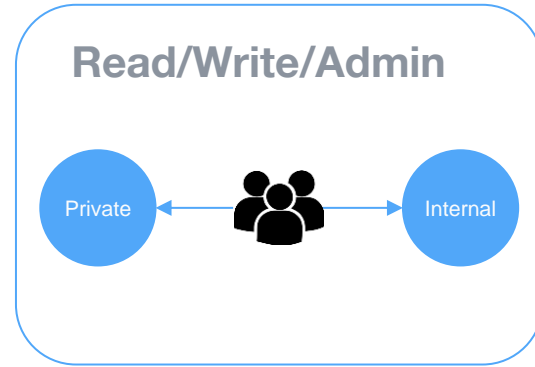
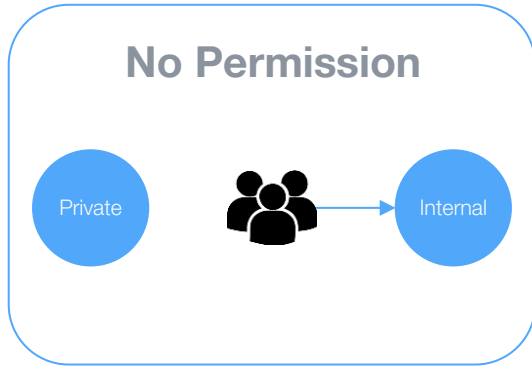


Choose a license

A license tells others what they can and can't do with your code. [Learn more](#).

Create repository

Repository visibility + base permission relationship



Typical enterprise scenario of **No Permission** with internal or private repositories

Roles

Role	Description
Read	Read-only access to Code and Actions. Can submit and comment on issues, pull requests, and discussions
Triage	Read-only permissions with the additional ability to manage issues, pull requests, discussions, assignments, and labels
Write	Gives write access to all parts of a repository project with the exception of the repository settings
Maintain	Ability to modify some settings of a repository including topics, enabling repository features, configuring merges and GitHub pages, pushing to protected branches
Admin	Has full administrative access to all features, settings and configurations of the repository project

GitHub Teams



Collaboration



Innersource



**Onboarding and
offboarding**



Security

Nested GitHub Teams

- Nested teams allow you to reflect your company's hierarchy within your org
- Parents team can have more than one child
 - Child teams inherit parent's permissions
 - Children receive parent's notifications
 - Users in a child team belong also to the parent team

40 teams in the octo-org organization

Employees

Engineering

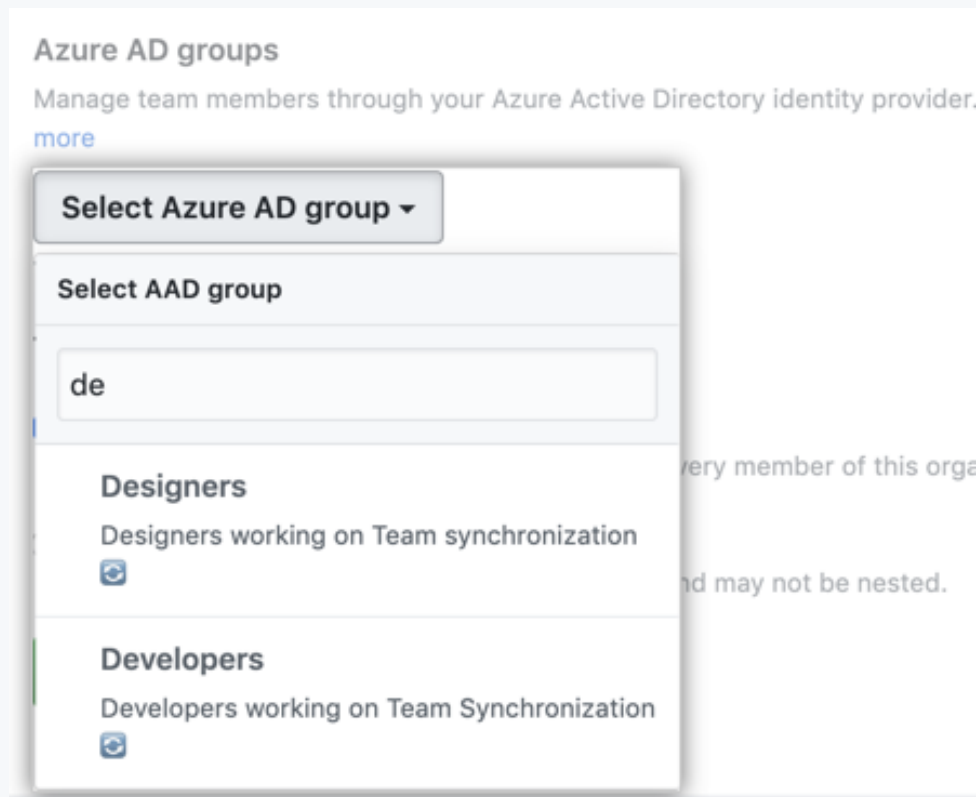
ApplicationEngineering

ClientSystems

Identity

Teams Synchronization

- Teams managed with Team Sync
 - Integrate IdP to synchronize groups to GitHub Teams
 - Manage rights and permissions in one place
 - Can't connect to parent team (if using nested teams)
- Teams managed in GitHub
 - Manage membership within GitHub
 - Keep them open, reduce friction



Security Managers in your Organization

Security managers Beta

Grant a team permission to manage security alerts and settings across your organization. This team will also be granted read access to all repositories. [Learn more about these security privileges.](#)

Q Search for teams

sec-man

×

- Security manager is an organization-level role that organization owners can assign to any team in an organization.
- It gives every member of the team permissions to view security alerts and manage settings for code security across your organization, as well as read permissions for all repositories in the organization.

GitHub Actions




- Fully integrated with GitHub
- Respond to any GitHub event
- Community-powered workflows
- Any platform, any language, any cloud

Starter workflows

- <https://github.com/actions/starter-workflows>

README.md



Starter Workflows

These are the workflow files for helping people get started with GitHub Actions. They're presented whenever you start to create a new GitHub Actions workflow.

If you want to get started with GitHub Actions, you can use these starter workflows by clicking the "Actions" tab in the repository where you want to create a workflow.

Untitled workflow
Setting up your workflow...

Get started with GitHub Actions

Choose a workflow to build, test, and deploy your code. Make code reviews, branch management, and issue triaging work the way you want.

Build and test your Ruby repository

Ruby Gem

Pushes a Ruby Gem to RubyGems and GitHub Packages Registry

Set up this workflow

gem install bundler

bundle install --jobs 4 --retry 3

bundle exec rake

Ruby


Build and test a Ruby project with Rake

Set up this workflow

Popular continuous integration workflows

Directory structure:

- **ci**: solutions for Continuous Integration
- **automation**: solutions for automating workflows.
- **icons**: svg icons for the relevant template



+ 96 contributors

Languages

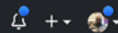
TypeScript 100.0%



Search or jump to...



[Pull requests](#) [Issues](#) [Codespaces](#) [Marketplace](#) [Explore](#)



[decyjphr / checks-bot](#) Private

[Unwatch](#) 1 [Star](#) 0 [Fork](#) 0

[Code](#) [Issues](#) [Pull requests](#) [Actions](#) [Projects](#) [Wiki](#) [Security](#) [Insights](#) [Settings](#)

Branch: master ▾

[Go to file](#)

[Add file ▾](#)

[Code ▾](#)



decyjphr committed a5e38e3 7 minutes ago

1 commits

1 branch

0 tags



demo.js

Create demo.js

7 minutes ago

Add a README with an overview of your project.

[Add a README](#)

About



No description, website, or topics provided.

Releases

No releases published
[Create a new release](#)

Packages

No packages published
[Publish your first package](#)

Languages

JavaScript 100.0%



Workflows

Workflow files glue together existing actions

- Listen for particular events
- Then run shell scripts
- Or pre-existing actions
- Actions run in VMs (Linux, Win, Mac)
 - Or Docker on Linux VM
- yaml syntax
- logs streaming & artifacts
- Secret store with each repository

```
1  name: Enforce repository settings
2
3  on: [push]
4
5  jobs:
6    probot-settings:
7
8      runs-on: ubuntu-latest
9
10     steps:
11       - uses: actions/checkout@v1
12       - name: Run probot-settings
13         uses: elstudio/actions-settings@v2-beta
14         env:
15           GITHUB_TOKEN: ${ secrets.GITHUB_TOKEN }}
16       - name: Remove workflow file from master branch
17         run: rm ../github/workflows/probot-settings.yml
18         if: endsWith(github.ref, '/master') && ! endsWith(git
19       - name: Commit changes
20         uses: elstudio/actions-js-build/commit@v2
21         env:
22           GITHUB_TOKEN: ${ secrets.GITHUB_TOKEN }}
23
```

Actions are reusable components

- Live in independent repositories
 - Public repositories for now
 - Official list: <https://github.com/actions>
- Written in JavaScript (node12)
 - May use GitHub Actions Toolkit JS for command line argument parsing, passing parameters, interacting with the GitHub API
- Or Docker
 - Similar to beta 1 actions, but with updated syntax & argument passing
 - Or point to existing actions published to Docker Hub

1

```
1 name: 'Wait'
2 description: 'Wait a designated number of milliseconds'
3 inputs:
4   milliseconds: # id of input
5     description: 'number of milliseconds to wait'
6     required: true
7     default: '1000'
8 outputs:
9   time: # output will be available to future steps
10    description: 'The message to output'
11 runs:
12   using: 'node12'
13   main: 'index.js'
```

2

```
1 name: 'GitHub Action to execute javascript build tools'
2 description: 'Executes npm install, followed by gulp or grun'
3 inputs:
4   wdPath:
5     description: 'Working directory path'
6     required: false
7     default: ''
8 runs:
9   using: 'docker'
10   image: 'Dockerfile'
11   entrypoint: 'entrypoint.sh'
12   env:
13     WD_PATH: ${ inputs.wdPath }
```


Security with self-hosted runners



Best practices for configuration of your runner nodes:

- Create a dedicated user for the Actions runner
- Enable limited `sudo`
- Run the Actions runner inside Docker



Self-hosted runners and Security

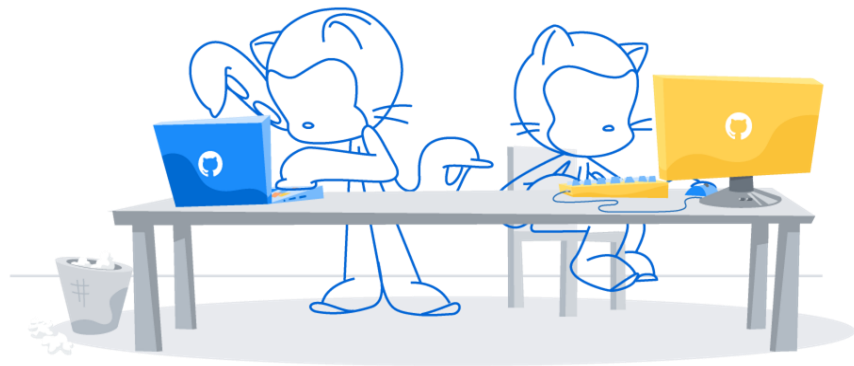
Many of your Actions may require *root* access in order to run. For example, if you are running an Action with Python natively, you may need to run *sudo* if you're checking out the `dschep/install-pipenv-action@v1` Action, as that runs *sudo* in order to install the necessary dependencies. In cases like this you may choose to avoid the Action, or you can limit the scope of your runner's permissions.

Security with self-hosted runners



Public repositories with self-hosted runners pose potential risks:

- Malicious programs running on the machine
- Escaping the machine's runner sandbox
- Exposing access to the machine's network
- Persisting unwanted or dangerous data on the machine



Self-hosted runners and **Security**

Forked repositories will contain the same Actions configuration as the parent repository, including the self-hosted runners. Creates the potential for a fork to run malicious code on a runner inside your network. For this reason, it is highly recommended to use self-hosted runners only with **private** repositories.

Secrets

Organization

- Allows secrets to be managed at Org level without duplication
- Effectively becomes Repository secrets
- Can be scoped to specific repositories

Repository

- Scoped to a repository
- Can be used to override Org secrets

Environment

- Scoped to a repository
- Can be used to override Repo secrets



● GitHub Secret store

- Built-in secret store
- Encrypted
 - LibSodium sealed box
- Use directly from your workflow
- Redacted in workflow logs
- API support
- Organization / repository / environment level secrets
- Can not read a secret value!

The screenshot shows the GitHub interface for managing secrets and variables. On the left is a sidebar with navigation options: General, Access (Collaborators and teams, Moderation options), Code and automation (Branches, Tags, Rules, Actions, Webhooks, Environments, Pages), Security (Code security and analysis, Deploy keys, Secrets and variables, Actions, Codespaces, Dependabot), and Integrations (GitHub Apps, Email notifications). The 'Secrets and variables' option is highlighted. The main content area is titled 'Actions secrets and variables' and contains explanatory text about secrets and variables, a 'New repository secret' button, and three sections: 'Environment secrets' (with a 'Manage environments' button and a secret 'MY_ENV_SECRET'), 'Repository secrets' (with a secret 'MY_REPO_SECRET'), and 'Organization secrets' (with a 'Manage organization secrets' button and a secret 'MY_ORG_SECRET').

General

Access

- Collaborators and teams
- Moderation options

Code and automation

- Branches
- Tags
- Rules Beta
- Actions
- Webhooks
- Environments
- Pages

Security

- Code security and analysis
- Deploy keys
- Secrets and variables**
- Actions
- Codespaces
- Dependabot

Integrations

- GitHub Apps
- Email notifications

Actions secrets and variables

Secrets and variables allow you to manage reusable configuration data. Secrets are **encrypted** and are used for sensitive data. [Learn more about encrypted secrets.](#) Variables are shown as plain text and are used for **non-sensitive** data. [Learn more about variables.](#)

Anyone with collaborator access to this repository can use these secrets and variables for actions. They are not passed to workflows that are triggered by a pull request from a fork.

Secrets **Variables** New repository secret

Environment secrets

Manage environments

MY_ENV_SECRET	Demo	Updated 3 minutes ago
---------------	----------------------	-----------------------

Repository secrets

MY_REPO_SECRET	Updated 2 minutes ago	Edit Delete
----------------	-----------------------	---

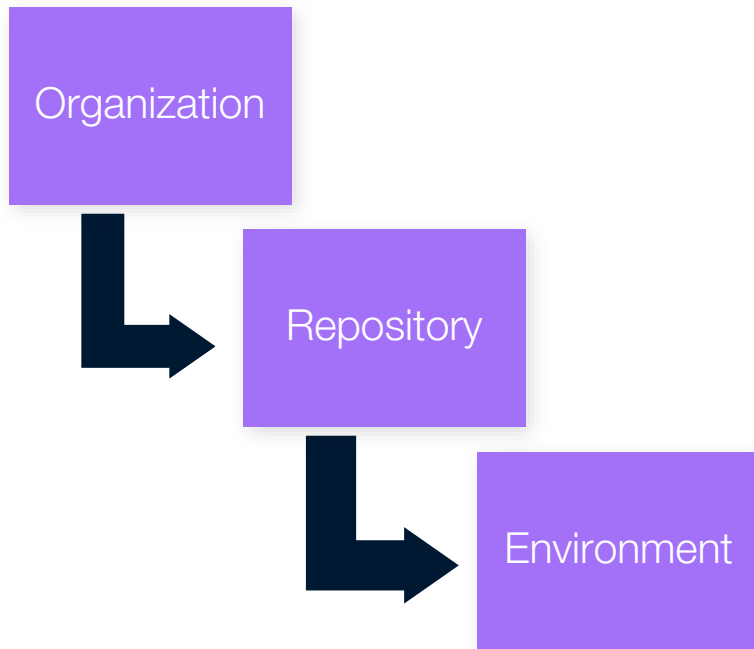
Organization secrets

Manage organization secrets

MY_ORG_SECRET	Updated on Apr 25
---------------	-------------------

Secrets

- Defined on org, repo, or environment level
- Secret context
 - `{{ secrets.MY_SECRET }}`
 - Set as input (`with:`) or environment (`env:`) for actions
- Set in UI or CLI
 - `$ gh secret set MY_SECRET -body "$value"`
 - `$ gh secret set MY_SECRET --env Prod`
 - `$ gh secret set MY_SECRET --org my-org`
- Masked in log



The GITHUB_TOKEN

`${{ secrets.GITHUB_TOKEN }}` or `${{ github.token }}`

Authenticate to GitHub to perform automation inside the workflow's repo

Default permission read/write for all scopes (old default) or set to read

```
permissions:  
  contents: read  
  pull-requests: write
```

```
permissions: read-all
```

```
permissions:  
  actions: read|write|none  
  checks: read|write|none  
  contents: read|write|none  
  deployments: read|write|none  
  issues: read|write|none  
  packages: read|write|none  
  pull-requests: read|write|none  
  repository-projects: read|write|none  
  security-events: read|write|none  
  statuses: read|write|none
```

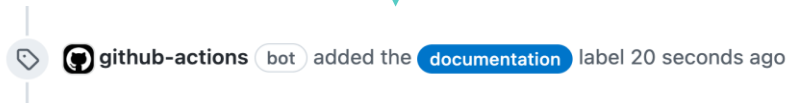
The `GITHUB_TOKEN`

Perform actions as “github-actions”:

```
permissions:  
  contents: read  
  issues: write  
  
label_issues:  
  runs-on: ubuntu-latest  
  if: github.event_name == 'issues'  
  
steps:  
  - uses: andymckay/labeler@e6c4322d0397f3240f0e7e30a33b5c5df2d39e90  
    with:  
      add-labels: documentation  
      repo-token: ${{ secrets.GITHUB_TOKEN }}
```

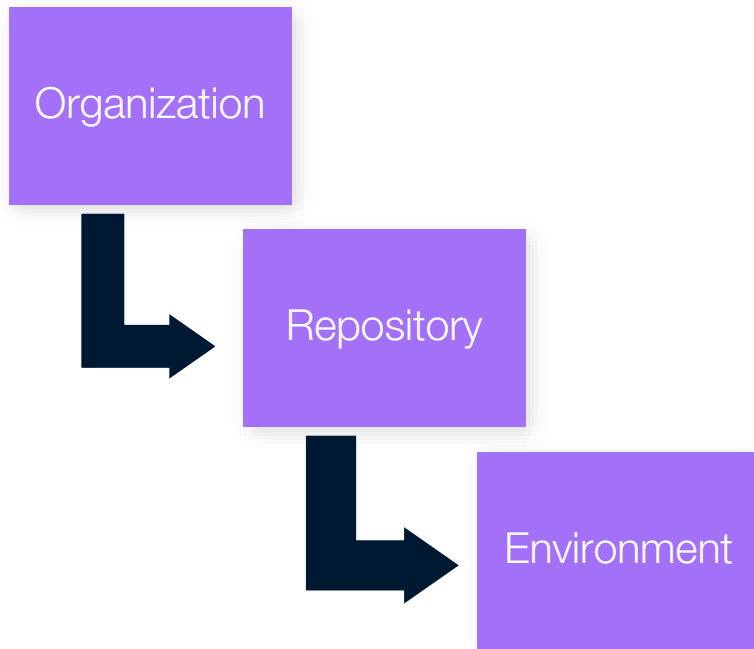


A teal arrow originates from the `issues: write` permission and points down to the `label_issues` section. Another teal arrow originates from the `repo-token` value in the `steps` section and points down to the GitHub Actions log entry.



Variables

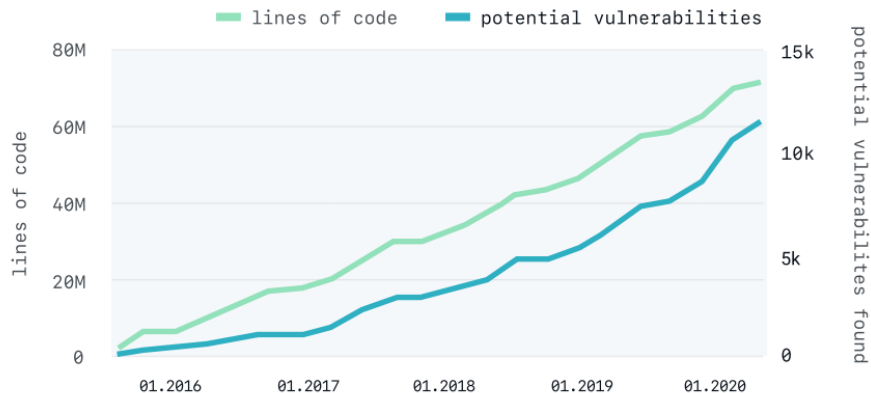
- Same setup as secrets, but no redacting
- Defined on org, repo, or environment level
- vars context
 - `${{ vars.MY_VAR }}`
 - Set as input (`with:`) or environment (`env:`) for actions
- **Not** masked in log



GHAS Application Security

The state of AppSec

Potential vulnerabilities found in source code scale with lines of code written



**Despite
billions of dollars
of investment...**

85% of applications still
contain a security issue

Code written in 2020 is just
as likely to introduce a
security issue as code
written in 2016

Flaws in applications are consistently the #1 attack vector for breaches

Source: Verizon Data Breach Investigations reports 2016, 2017, 2018, 2019 and 2020.

The state of AppSec

Is falling further behind the current state of Development



1:100 Security team members to developers



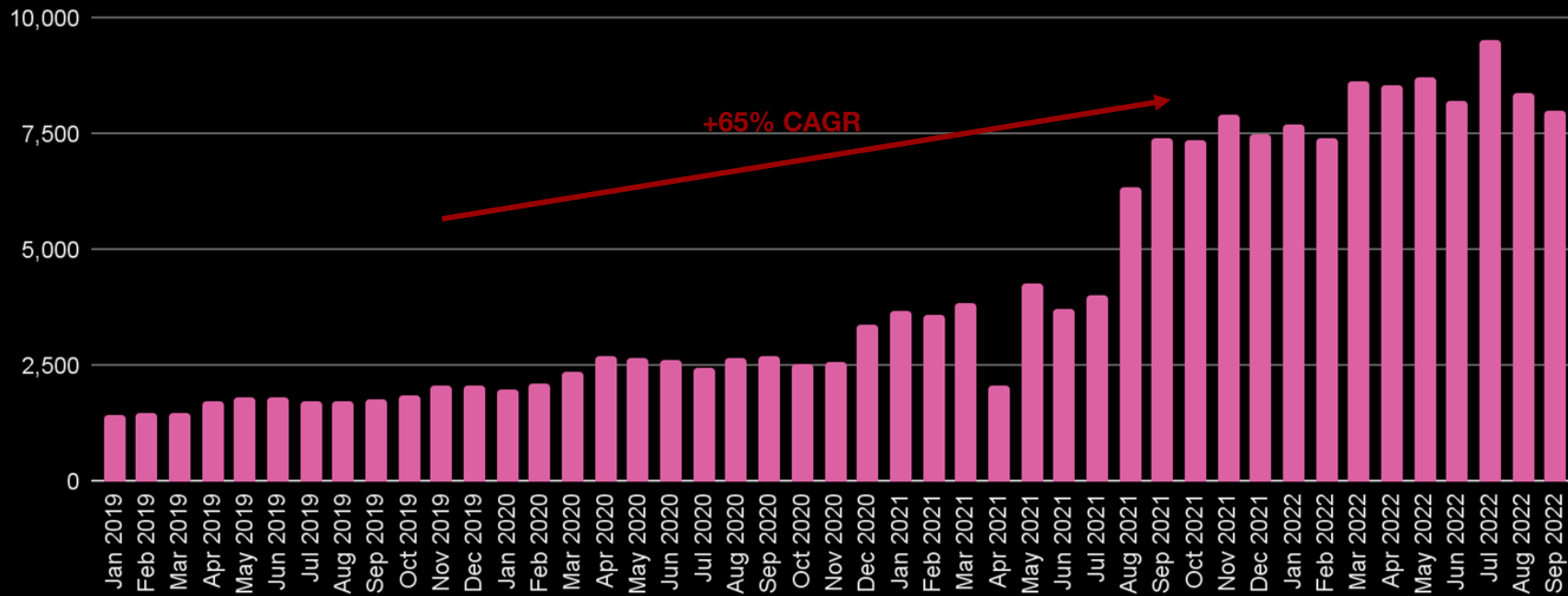
Lack of knowledge voted the main AppSec challenge



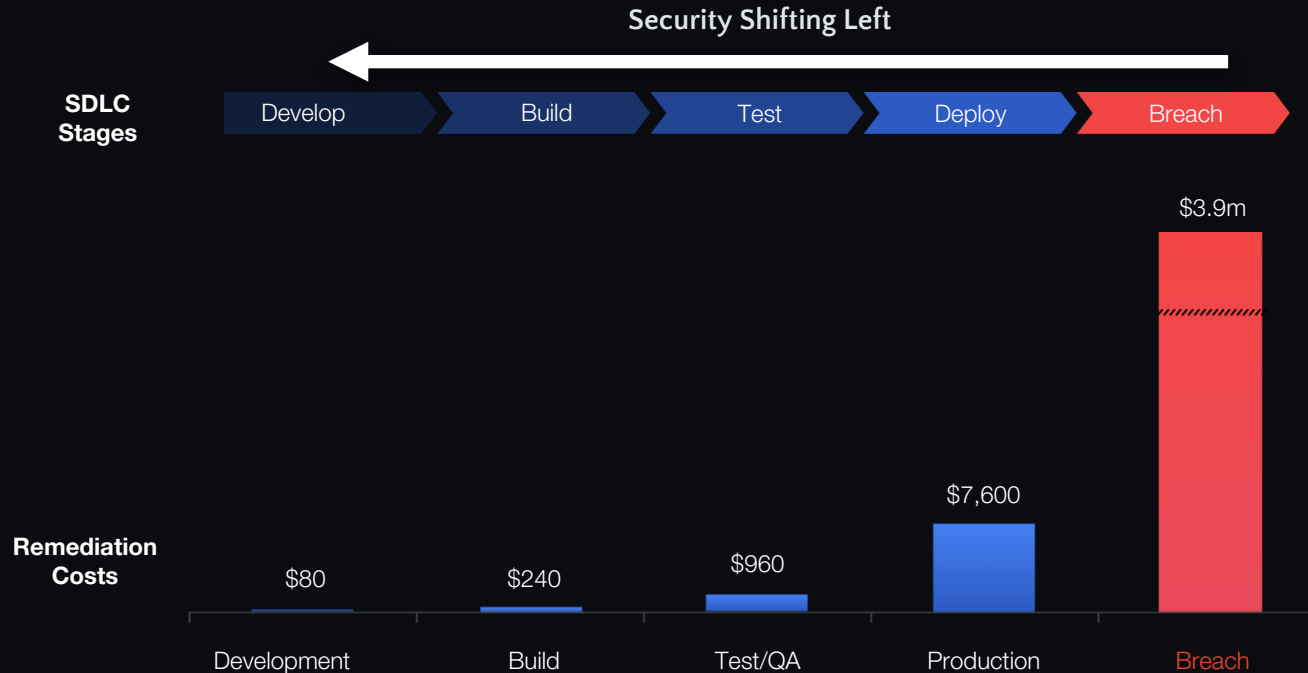
Remediation trends are stagnant

We're seeing more credential leaks than ever

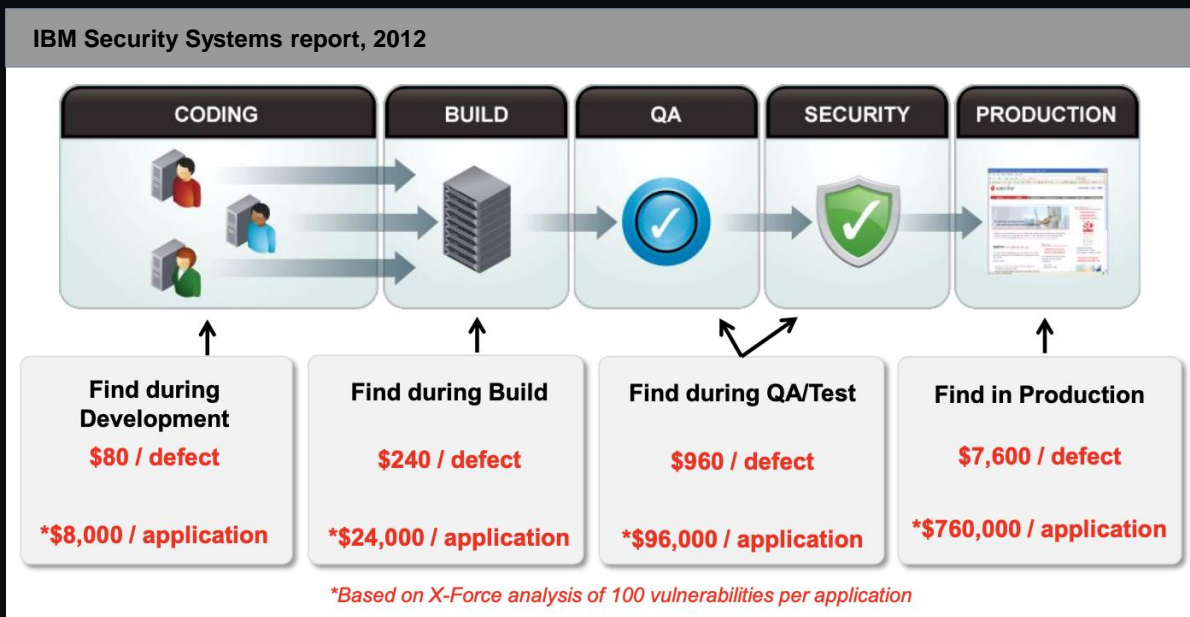
GitHub access tokens leaked in public repositories



Everyone wants to shift security left...

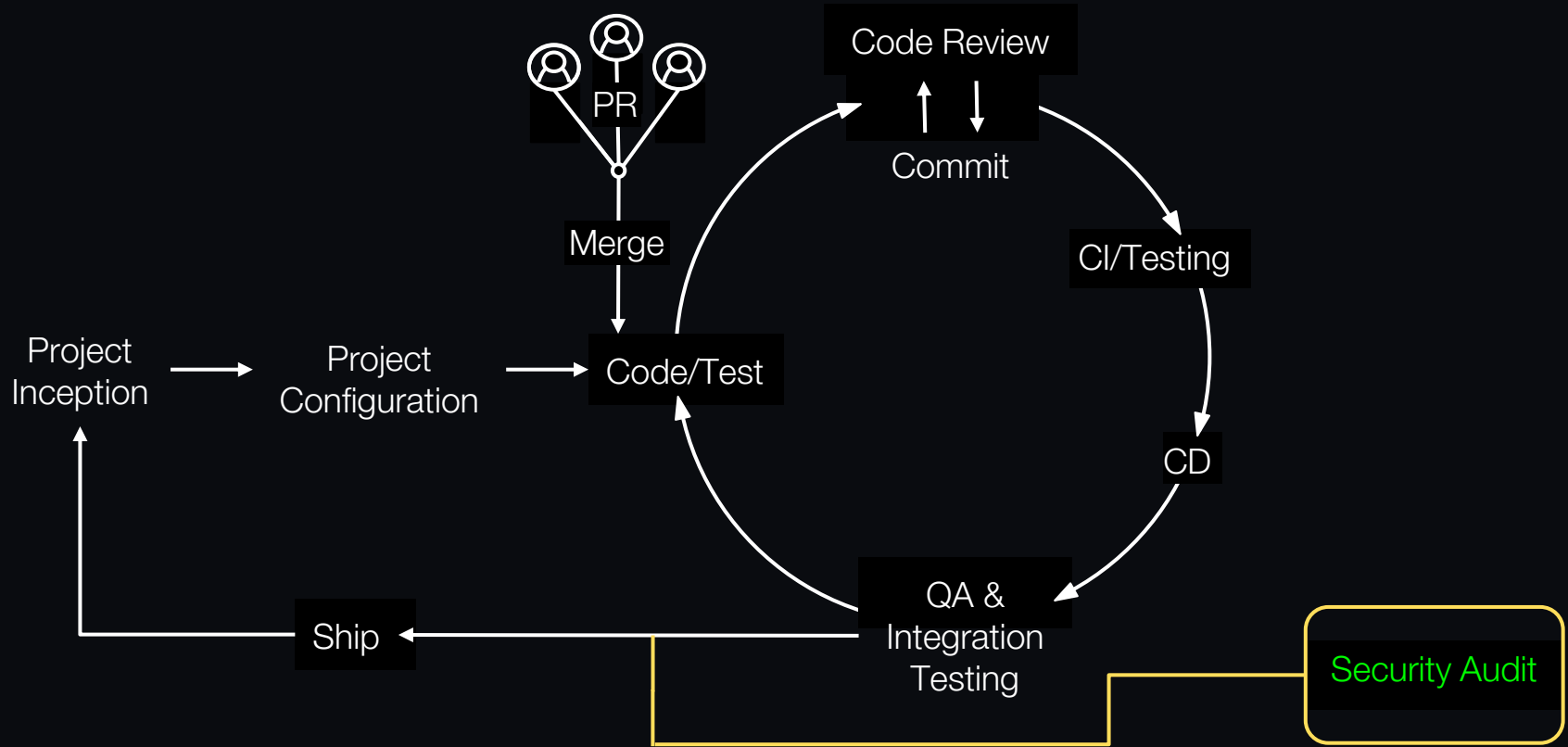


... but the industry has been trying to shift left for at least a decade

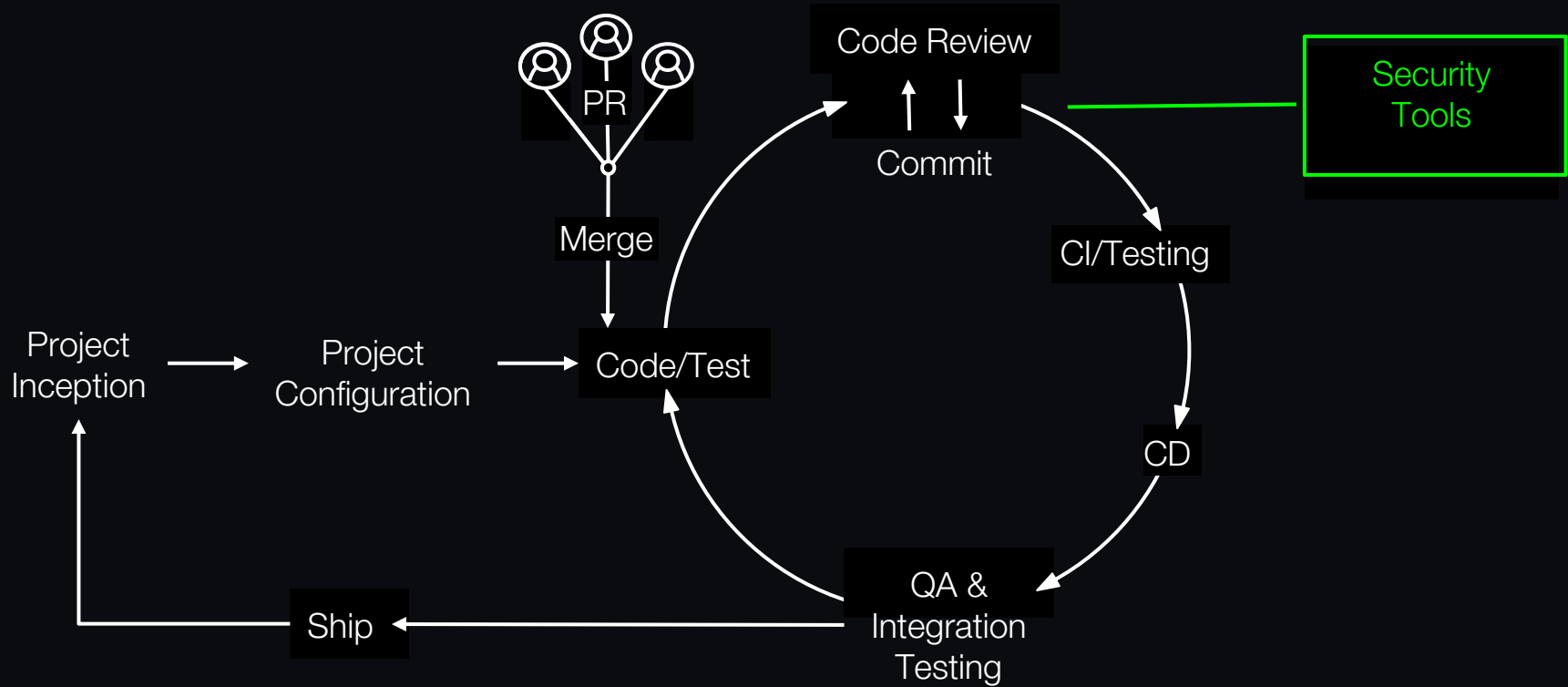


*GitHub believes that making this shift
requires a developer-first approach to
all our security products*

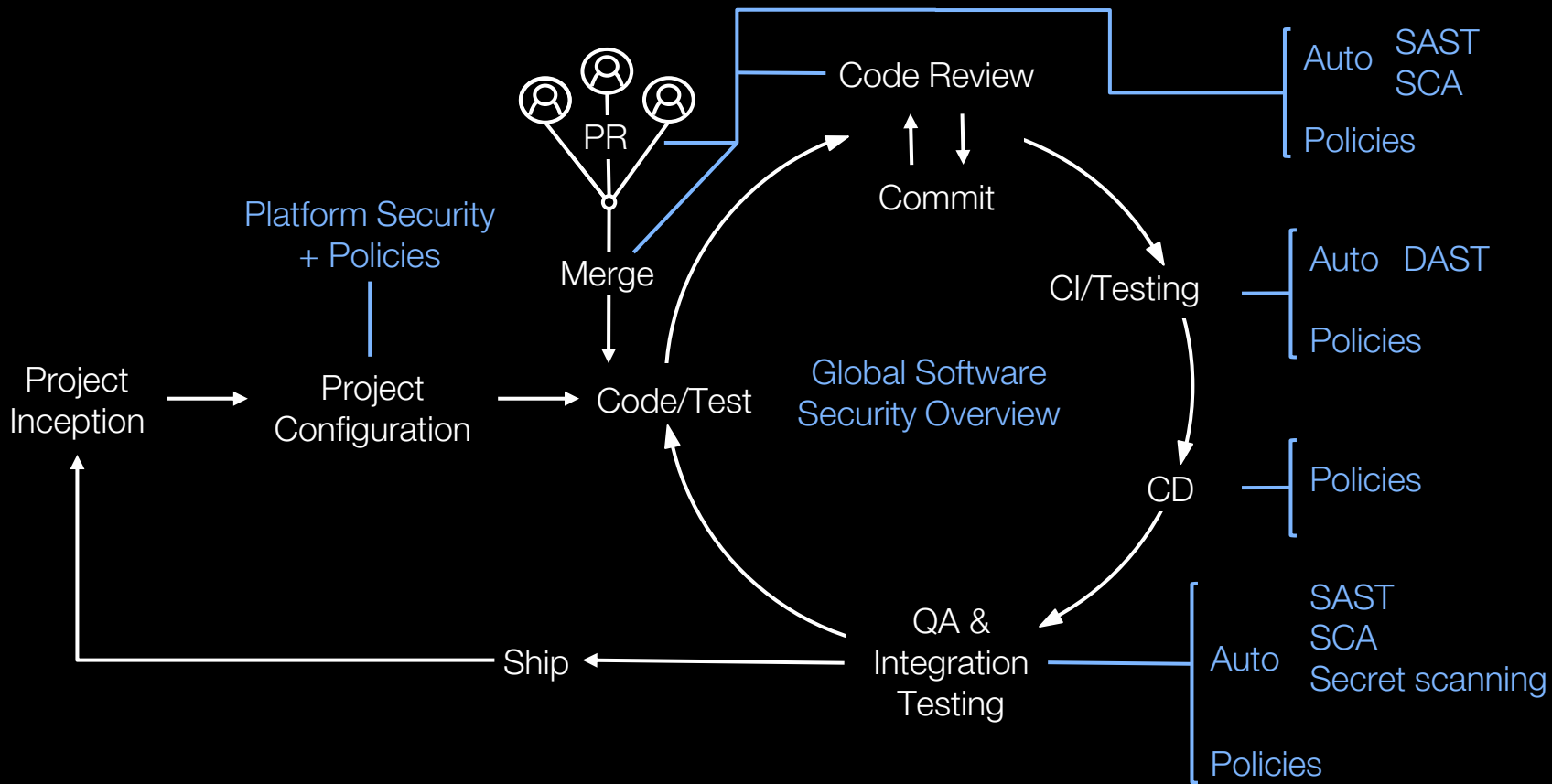
Basic Application Security scenario



Application Security scenario



Application Security - Targeted state



Developer first?

We see three key aspects to being a “developer first” tool:

Integrate *directly* into the developer workflow.

Make setup and deployment fast and easy.

Produce high quality results with low numbers of false positives.

GitHub Advanced Security: Current capabilities



supply
chain



code



platform

• **Dependency graph**

View your dependencies

• **Advisory database**

Canonical database of
dependency vulnerabilities

• **Security alerts and updates**

Notifications for vulnerabilities in
your dependencies, and pull
requests to fix them

• **Dependency review**

Identify new dependencies and
vulnerabilities in a PR

• **Secret scanning**

Find API tokens or other secrets
exposed anywhere in your git
history.

• **Code scanning**

Static analysis of every git push,
integrated into the developer
workflow and powered by
CodeQL

• **Branch protection**

Enforce requirement for pushing
to a branch or merging PRs

• **Commit signing**

Enforce requirement that all
commits are signed

• **Security overview**

View security results of all kinds across
your organization

Dependabot

- Developers (and others!) notified by an alert when new vulnerable dependencies are detected.
- Automatically open pull requests to fix dependency vulnerabilities.
- Supports dependency review within PRs to prevent adding known vulnerable dependencies.

The screenshot shows a GitHub pull request titled "Bump axios from 0.18.0 to 0.18.1 in /frontend #4". The pull request is created by the Dependabot bot and targets the master branch. A yellow banner at the top states: "This automated pull request fixes a security vulnerability" with a link to learn more about Dependabot security updates. The pull request description includes the following information:


- Release notes:** Sourced from [axios's releases](#).
- v0.18.1 Security Fix:**
 - Destroy stream on exceeding maxContentLength (fixes [#1098](#) ([#1485](#)) - Gadzhi Gadzhiev)
- Changelog**
- Commits**
- compatibility:** 99%
- Dependabot will resolve any conflicts with this PR as long as you don't alter it yourself. You can also trigger a rebase manually by commenting `@dependabot rebase`.**
- Dependabot commands and options**

The pull request is reviewed by [cmbolling](#) and is currently in progress. The right sidebar shows the "Reviewers" section with a "Request" button, "Assignees" (No one—assign yourself), "Labels" (dependencies, javascript), "Projects" (None yet), "Milestone" (No milestone), "Linked issues" (Successfully merging this pull request may close these issues. None yet), and "Notifications" (Subscribe).

Secret scanning

- Identify secrets across your entire git history with high accuracy.
- [Push protection](#) - prevent secrets from being pushed to GitHub.
- Developers (and others!) notified by an alert if secrets are pushed.
- Automated revocation for public repositories, private repositories include a review workflow.

```
1 namespace DataModel
2 {
3     public static class LoginHelper
4     {
5         public static String ServiceUrl = "https://cloud.example.com";
6         public static String ClientID = "DataModel-0001";
7         public static String ClientSecret = "A002019DRBES$%FA
8         public static string RedirectURL = "https://example.com/redirect";
9
10        /// <summary>
11        /// Handles acquiring all relevant tokens for the app
12        /// </summary>
13        /// <returns> Asynchronous task </returns>
14        /// </summary>
15        /// Handles acquiring all relevant tokens for the app
```



Code scanning

- Find vulnerabilities before they are merged into the code base with automated CodeQL scans
- Integrate results directly into the developer workflow
- Run custom queries and the community-powered GitHub query set
- Extensible, with support for other SAST tools

The screenshot displays the GitHub Code Scanning interface for a repository named 'dsp-testing / code-scanning-demo'. The left sidebar shows navigation options: Overview, Security policy, Security advisories (0), Dependabot alerts (0), Code scanning alerts (1), CodeQL, and Detected secrets (0). The main content area is titled 'Server-side URL redirect' with a 'Beta' label and a 'Give us feedback' link. Below the title, a description states: 'Server-side URL redirection based on unvalidated user input may cause redirection to malicious web sites.' There are three status buttons: 'Open' (green), 'Warning' (yellow), and 'CWE-601' (blue). A 'security' tag is also present. The code snippet is from 'test.ts' on the 'master' branch, showing a function 'sendRedirect' that sets a 'Location' header based on user input. A yellow highlight indicates a vulnerability: 'Untrusted URL redirection due to user-provided value.' Below the code, a table lists the tool (CodeQL), rule ID (js/server-side-unvalidated-url-redirection), and query (View source). The table also includes a detailed description of the vulnerability: 'Directly incorporating user input into a URL redirect request without validating the input can facilitate phishing attacks. In these attacks, unsuspecting users can be redirected to a malicious site that looks very similar to the real site they intend to visit, which is controlled by the attacker.'

Tool	Rule ID	Query
CodeQL	js/server-side-unvalidated-url-redirection	View source

Directly incorporating user input into a URL redirect request without validating the input can facilitate phishing attacks. In these attacks, unsuspecting users can be redirected to a malicious site that looks very similar to the real site they intend to visit, which is controlled by the attacker.

Reviewing Alerts

Overview

Repositories

Projects

Packages

Teams

People

Security

Security

Overview

Risk

Coverage

Metrics

Secret scanning

Alerts

Dependabot

Code scanning

Secret scanning

You can only see data from repositories for which you have [permission](#) to view.

Overview

Alert trends and insights across your organization.

Dec 15, 2023 - Jan 14, 2024

Filter

Try modifying your filters to see the security impact on your organization.



Monitoring and responding to alerts

Code samples for "List code scanning alerts for an organization"

Request example

GET /orgs/{org}/code-scanning/alerts

cURLJavaScriptGitHub CLI

```
// Octokit.js
// https://github.com/octokit/core.js#readme
const octokit = new Octokit({
  auth: 'YOUR-TOKEN'
})

await octokit.request('GET /orgs/{org}/code-scanning/alerts', {
  org: 'ORG',
  headers: {
    'X-GitHub-API-Version': '2022-11-28'
  }
})
```



Q&A