

# GitHub Advanced Security for Azure DevOps

## Configure and Enable Core Security Features



vighnesh-  
notes

[vighneshnaik.com](https://vighneshnaik.com)

[linkedin.com/in/vighneshnaik6](https://linkedin.com/in/vighneshnaik6)

### WHAT IS IT?



GitHub Advanced Security for Azure DevOps brings GitHub Advanced Security capabilities to **Azure Repos** in **Azure DevOps Services**.

It helps you find and fix security issues early in your development lifecycle.

### CORE FEATURES



#### Secret Scanning (push protection)

Prevents commits that contain secrets (such as API keys or tokens) from being pushed to your repository.



#### Secret Scanning (repository scanning)

Scans your existing repositories for secrets and notifies you of any issues found.



#### Dependency Scanning

Finds vulnerable open source dependencies in your code.



#### Code Scanning with CodeQL

Analyzes your code for security vulnerabilities using CodeQL and provides actionable results.

### PRODUCTS

Available in Azure DevOps as:



#### GitHub Secret Protection for Azure DevOps

Includes secret scanning (push protection and repository scanning).



#### GitHub Code Security for Azure DevOps

Includes dependency scanning and code scanning with CodeQL.

### WHERE IT WORKS



- ✓ Azure DevOps Services only.
- ✓ For code Git repositories in Azure Repos.
- ✓ Not available for Azure DevOps Server (on-premises).

### HOW IT WORKS



#### Code

You write and push code to Azure Repos (Git repositories).



#### Scan

Security features analyze your code, commits and dependencies.



#### Alert

Issues are detected and surfaced in Azure DevOps.



#### Fix

You fix the issues in your code and dependencies.



#### Protect

Your code is more secure and your organization is safer.

### WHY IT MATTERS



#### Prevent secret leaks

Stop sensitive information from being exposed in your code.



#### Find vulnerable dependencies

Identify and remediate risky open source packages early.



#### Detect code vulnerabilities

Find security issues in your custom code with CodeQL.



#### Improve security visibility

Get clear, actionable insights to improve the overall security posture.

### GET STARTED



Start by enabling the right **protection level** for each repository.



Choose the features that fit your team's needs and gradually enable them across your repositories.



Stay consistent, practice hands-on, and think like a DevOps engineer.  
Build. Test. Learn. Repeat.



You've got this!





# Prerequisites & Permissions

## What you need before enabling Advanced Security



vighnesh-  
notes

[vighneshnaik.com](https://vighneshnaik.com)

[linkedin.com/in/vighneshnaik6](https://linkedin.com/in/vighneshnaik6)

### PERMISSIONS OVERVIEW

Different roles have different capabilities in GitHub Advanced Security for Azure DevOps.



#### Contributor

Can view a repository summary of alerts (security findings).



#### Project Administrator

Can dismiss alerts.



#### Manage Settings

To manage security settings, you must be a member of the **Project Collection Administrators** group OR have the **Advanced Security manage settings** permission.

### SUPPORTED ENVIRONMENT



#### Azure DevOps Services only

GitHub Advanced Security for Azure DevOps is available in Azure DevOps Services.



#### Azure Repos Git repositories only

Supported for Azure Repos using Git repositories.



#### Not available for Azure DevOps Server (on-premises)

GitHub Advanced Security is not supported on Azure DevOps Server or on-premises deployments.

### SELF-HOSTED AGENTS - EXTRA PREREQUISITES

If you use self-hosted agents for CodeQL scanning, ensure the following:



#### Allow list required domains

Allow access to the following domains in your network:

governance.dev.azure.com  
dev.azure.com  
advsec.dev.azure.com  
and their visualstudio.com  
equivalents:  
governance.visualstudio.com  
dev.visualstudio.com  
advsec.visualstudio.com



#### Compatible .NET runtime

Ensure a compatible .NET 8.x runtime is installed on your self-hosted agents.

**.NET 8.x**



#### CodeQL bundle

Ensure the CodeQL bundle is installed on your agents or allow automatic installation during the scan run.



You can pre-install the CodeQL bundle or let the pipeline install it automatically.

### QUICK CHECKLIST

Use this checklist to confirm you are ready.



#### Repository type

Using Azure Repos Git repositories.



#### Permissions

You have the required permissions (view, dismiss or manage settings).



#### Network allow list

Required domains are allowed (for self-hosted agents).



#### .NET runtime

Compatible .NET 8.x runtime is available (on self-hosted agents).



#### Agent readiness

CodeQL bundle is installed or can be installed automatically.

### TIP



Check your permissions and networking first to avoid onboarding issues later!

### WHY IT MATTERS



Ensures a smooth and successful enablement experience.



Prevents common errors and scan failures.



Helps your team get value from Advanced Security quickly.



Stay consistent, practice hands-on, and think like a DevOps engineer.  
**Build. Test. Learn. Repeat.**



**You've got this!**





# Enable Advanced Security

## Repository, Project, and Organization onboarding

Enable Secret Protection or Code Security at the right scope



vighnesh-  
notes

vighneshnaik.com

linkedin.com/in/  
vighneshnaik6

### REPOSITORY-LEVEL ONBOARDING



Enable Advanced Security for a single repository.

You can enable either **Secret Protection** or **Code Security** (or both) for a repository.

- 1 Go to Settings**  
Open your repository and go to Settings.
- 2 Select Security**  
In the left menu, choose Advanced Security.
- 3 Toggle features**  
Enable Secret Protection and/or Code Security.
- 4 Review estimate**  
Review the estimated cost based on active committers.
- 5 Begin billing**  
Click Begin billing to activate.

### PROJECT-LEVEL ONBOARDING



Enable Advanced Security for all repositories in a project.

You can enable either **Secret Protection** or **Code Security** (or both) for a project.

- 1 Go to Settings**  
Open your project and go to Settings.
- 2 Select Repositories**  
Choose Repositories from the left menu.
- 3 Toggle features**  
Enable Secret Protection and/or Code Security.
- 4 Review estimate**  
Review the estimated cost based on active committers across the project repositories.
- 5 Enable all (existing repos)**  
Click Enable all to apply to existing repositories in this project.
- 6 Optional: Auto-enable**  
Turn on auto-enable for new repositories in this project (optional).
- 7 Begin billing**  
Click Begin billing to activate.

### ORGANIZATION-LEVEL ONBOARDING



Enable Advanced Security for all repositories in your organization.

You can enable either **Secret Protection** or **Code Security** (or both) for your organization.

- 1 Go to Settings**  
Go to your organization and open Settings.
- 2 Select Repositories**  
Choose Repositories from the left menu.
- 3 Toggle features**  
Enable Secret Protection and/or Code Security.
- 4 Review estimate**  
Review the estimated cost based on active committers across the organization.
- 5 Enable all (existing repos)**  
Click Enable all to apply to existing repositories in this organization.
- 6 Optional: Auto-enable**  
Turn on auto-enable for new repositories in this organization (optional).
- 7 Begin billing**  
Click Begin billing to activate.

### HOW IT WORKS



Go to Settings



Select Repos/Repositories



Choose the level (Repository, Project, or Organization)



Toggle Secret Protection and/or Code Security



Review estimate for active committers



Enable all (for existing repos where relevant)



Begin billing to activate



Optionally enable auto-enable for new repositories or projects

### KEY THING TO REMEMBER



Enable all applies to **existing repositories** in the selected scope. The **auto-enable** toggle is separate and only affects **future repositories or projects**.

### VISUAL INDICATOR



A **shield icon** appears next to repositories with Advanced Security enabled (Secret Protection and/or Code Security).



my-awesome-repo



Protected



Stay consistent, practice hands-on, and think like a DevOps engineer.  
Build. Test. Learn. Repeat.



You've got this!





# Secret Protection Setup

## Push protection and repository secret scanning

Keep your code and secrets safe in Azure DevOps



vighnesh-  
notes

[vighneshnaik.com](https://vighneshnaik.com)

[linkedin.com/in/vighneshnaik6](https://linkedin.com/in/vighneshnaik6)

### WHAT SECRET PROTECTION INCLUDES



#### Push Protection

Checks for secrets when you push code and blocks the push if secrets are detected.



#### Secret Scanning Alerts and Notifications

Scans your repositories for secrets and notifies you of any issues found.



#### Security Overview

Provides a centralized view of secret protection status, alerts and remediation guidance.

### WHAT HAPPENS WHEN ENABLED



#### Push Protection and Repository Scanning are Automatically Enabled

When you enable secret protection, both push protection and repository scanning are turned on for the repository.



#### Repository Scanning Starts in the Background

Azure DevOps starts scanning your repository in the background to find existing secrets.



#### No Additional Setup Required

Scans run automatically, so you can quickly get visibility into potential exposures.

### IMPORTANT BEHAVIOR



#### Push Protection Checks Pushes After Enablement

Push protection only checks new pushes made after you enable secret protection. It blocks pushes that contain detected secrets.



#### Repository Scanning Can Detect Existing Secrets

Repository scanning can find secrets that already exist in your repository, including in historical commits.

### WHERE TO CONTROL IT



#### Repository Settings Page

You can enable or disable push protection from your repository settings in Azure DevOps.

Azure DevOps > Repos > Repository Settings  
> Secret Protection

### WHY IT MATTERS



#### Prevents Secrets from Leaking

Helps stop credentials, API keys and tokens from being accidentally committed and exposed in your code.



#### Surfaces Existing Exposures Early

Repository scanning helps you find secrets already in your code (including historical commits) so you can fix them before they are misused.



Stay consistent, practice hands-on, and think like a DevOps engineer.  
Build. Test. Learn. Repeat.



You've got this!





# Dependency Scanning

## Find vulnerable open source dependencies



vighnesh-  
notes

[vighneshnaik.com](https://vighneshnaik.com)

[linkedin.com/in/vighneshnaik6](https://linkedin.com/in/vighneshnaik6)

### WHAT IS DEPENDENCY SCANNING?



Dependency Scanning finds **vulnerable open source dependencies** in your code.

It checks the packages and libraries your application uses and identifies known security vulnerabilities so you can fix them early.

### REQUIRES CODE SECURITY



Dependency scanning is a feature of **GitHub Code Security** for Azure DevOps.

You must enable Code Security in your Azure DevOps organization to use dependency scanning.

### PIPELINE-BASED SCANNING



Dependency scanning runs in your pipelines.

Results from pipeline runs are **aggregated per repository**, giving you a single, unified view of dependency vulnerabilities for each repository.

### DEFAULT SETUP (DEFAULT BRANCH)



Use the repository setting **"Scan for vulnerable dependencies"**

This automatically includes the dependency scanning task in pipelines that target the default branch and in pull request (PR) builds to the default branch.

### ADVANCED SETUP



For more control, add the **dependency scanning task** to **all pipelines** you want scanned.

This is useful if you want to scan all branches or have custom pipeline configurations.

### OUTPUTS & BENEFITS



**Dependency alerts**

Get notified about vulnerable packages in your dependencies.



**Risk visibility**

Understand the security risks in your open source dependencies.



**Earlier remediation**

Find and fix vulnerable dependencies early in your development lifecycle.

### HOW IT WORKS



**1. Enable Code Security**

Turn on Code Security in your organization.



**2. Choose setup option**

Use the default branch setting or add the task to your pipelines (advanced).



**3. Scan pipeline**

Run your pipelines. Dependency scanning checks your dependencies.



**4. Review alerts**

View aggregated results per repository in the Security view.



**5. Fix dependencies**

Update vulnerable packages to secure versions.

### WHY IT MATTERS



**Keep your software safe**  
Reduce the risk from known vulnerable packages.



**Build with confidence**  
Use trusted, up-to-date dependencies.

### GET STARTED



**Enable Code Security** and set up dependency scanning for your repositories.



A small setup step can help you find big security risks in your dependencies!



Stay consistent, practice hands-on, and think like a DevOps engineer.  
**Build. Test. Learn. Repeat.**



**You've got this!**





# Code Scanning with CodeQL

Default setup, advanced setup, and Copilot Autofix



vighnesh-  
notes

vighneshnaik.com

linkedin.com/in/  
vighneshnaik6

## REQUIRES CODE SECURITY



Code scanning with CodeQL is part of GitHub Advanced Security for Azure DevOps, so it requires **Code Security** to be enabled.

Make sure Code Security is enabled at the organization or project level before you can use code scanning.

## PIPELINE-BASED SCANNING



Code



Azure  
Pipelines



Results

Code scanning with CodeQL is pipeline-based. It analyzes your code during pipeline runs, and the results are aggregated per repository so you get a unified view of security issues.

## DEFAULT SETUP

Quickest option to get started



Runs on a schedule using Azure Pipelines.



Automatically detects supported languages in your repository.



Updates automatically if your repository languages change.



Uses a shared organization-level agent pool and schedule.

### SUPPORTED BY DEFAULT SETUP



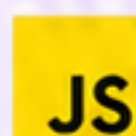
C#



C/C++



Java



JavaScript/  
TypeScript



Python



Ruby



Rust



Go and Swift are **not supported by default setup**. Use advanced setup for Go and Swift.



Go



Swift

## ADVANCED SETUP

Full control with CodeQL pipeline tasks



Add CodeQL pipeline tasks directly in your Azure Pipelines YAML for full control over your configuration.



Customize build steps to match your project needs.



Run code scanning on multiple branches (e.g., main, develop, release/\*).



Use different agent pools (self-hosted or Microsoft-hosted).



Configure additional options and settings to tailor the analysis to your environment.

## BEST-PRACTICE TIP



**Start with default setup**

It's the fastest and easiest way to get value.

**Move to advanced setup**

when you need more control, such as custom build steps, multiple branches, or different agent pools.

## COPILOT AUTOFIX

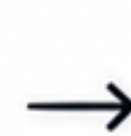


Limited public preview

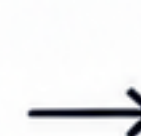
Copilot Autofix works only for CodeQL-based code scanning alerts. It analyzes the issue and creates a pull request with a proposed fix.



CodeQL  
alert



Copilot  
Autofix



Pull request  
with fix



**Copilot Autofix does not automatically remediate issues.**

You still need to review the proposed changes, validate the fix, and merge the pull request.



Stay consistent, practice hands-on, and think like a DevOps engineer.  
Build. Test. Learn. Repeat.



You've got this!





# Pull Request Security Checks

## Annotations, status checks, and branch policies



### PULL REQUEST ANNOTATIONS

Annotations for dependency scanning and code scanning appear automatically in your pull request.



- ✓ Annotations are created automatically when a **build validation policy** exists and the relevant GitHub Advanced Security tasks are included.
- ✓ Both dependency scanning and code scanning can add annotations to your PR.
- ✓ Scans must run on the **default/target branches** before the **source PR branch** for annotations to appear.



### STATUS CHECKS OVERVIEW

GitHub Advanced Security creates two status checks for pull requests.



#### AdvancedSecurity/AllHighAndCritical

- Blocks the pull request on all critical and high vulnerabilities across the repository (existing and new).
- Helps prevent vulnerable code from being merged.



#### AdvancedSecurity/NewHighAndCritical

- Blocks the pull request on new critical and high vulnerabilities only.
- Requires a build validation policy with Advanced Security tasks that scan the PR branch.

### HOW TO CONFIGURE AS BRANCH POLICIES

Add the GitHub Advanced Security status checks as branch policies in Azure DevOps.



1. Go to Project settings



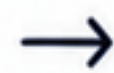
2. Select Repos



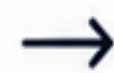
3. Go to Policies



4. Choose the protected branch (e.g., main)



5. Add Build validation



6. Add the status check(s) (AdvancedSecurity/AllHighAndCritical and/or AdvancedSecurity/NewHighAndCritical)



After the first successful pipeline run, the status check options (AdvancedSecurity/AllHighAndCritical and AdvancedSecurity/NewHighAndCritical) will appear in the list.

### IMPORTANT PIPELINE NOTE

If multiple GitHub Advanced Security tasks exist in your pipeline, enable **"Wait for Processing"** on the following tasks:



AdvancedSecurity-CodeQL-Analyze



AdvancedSecurity-Publish

This ensures the analysis completes and results are available for status checks and annotations.

### CAUTION



Leave the advanced options at their default values when adding the build validation policy.

Changing the authorized identity or requiring an iteration ID can prevent the status checks from posting to your pull request.

Use the default settings to ensure status checks work correctly.





# Implementation Playbook & Key Takeaways

## How to roll it out successfully



vighnesh-  
notes

vighneshnaik.com  
linkedin.com/in/  
vighneshnaik6

### 1. START SMALL



- Begin with one or a few repositories to pilot GitHub Advanced Security for Azure DevOps.
- Learn, fine-tune, and build confidence before rolling it out to more repositories.

Start small. Prove value.  
Then scale.

### 2. CHOOSE THE RIGHT PRODUCTS



#### Secret Protection

Find and prevent secrets from being exposed in your code.



#### Code Security

Find vulnerable dependencies and security issues in your code using CodeQL.

Choose Secret Protection, Code Security, or both, based on your team's needs.

### 3. VALIDATE PREREQUISITES



- Ensure you have the required permissions (e.g., Project Collection Administrator or security administration rights).
- Verify your organization is in Azure DevOps Services (not on-premises).
- Confirm you are using Azure Repos (Git).

Get the basics right before you enable.

### 4. SET UP CODEQL



- Start with the default CodeQL setup for quick and easy enablement.
- The default setup works well for most repositories.
- Move to advanced setup (fully customizable) only when you need more control (e.g., custom build steps, languages, or configuration).

Start simple. Go advanced when needed.

### 5. ENFORCE SECURE MERGES



#### Build Validation

Run CodeQL analysis in your pipelines.



#### PR Annotations

Show security findings directly in pull requests.



#### Status Checks

Require successful security checks before allowing merges.

Make security a natural part of the pull request process.

### 6. REVIEW AND FIX ALERTS



- Review alerts regularly in the Security tab.
- Fix exposed secrets (rotate and remove).
- Remediate vulnerable dependencies (update packages).
- Address CodeQL code findings (review and fix the issues).

Find it. Fix it. Keep it clean.

### 7. SELF-HOSTED AGENTS



- If you use self-hosted agents, make sure they are ready.
- Verify network connectivity to GitHub and required endpoints (for CodeQL downloads).
- Ensure the required runtime environments are installed (e.g., Java, .NET, etc.).
- Confirm the CodeQL bundle can be downloaded and used on your agents.

A well-prepared agent keeps your scans running smoothly.

### 8. KNOW THE SCOPE AND LIMITS



- Available for Azure DevOps Services only (not on-premises).
- Works with Azure Repos Git repositories only (not TFVC).
- Copilot Autofix for code scanning is in public preview and has limited language and repository support.
- Feature set and limits may evolve — check the latest documentation.

Know the boundaries to set clear expectations.

### ROLL OUT CHECKLIST

- ✓ Start with one or a few repositories.
- ✓ Choose Secret Protection, Code Security, or both.
- ✓ Validate prerequisites and permissions.
- ✓ Enable CodeQL (default setup first).
- ✓ Set up build validation, PR annotations, and status checks.
- ✓ Review and fix alerts regularly.
- ✓ Prepare self-hosted agents (if used).
- ✓ Understand scope and limits.
- ✓ Communicate and scale to more repos.

### BEST PRACTICES



- ✓ Integrate security early in your development lifecycle.
- ✓ Keep repositories, dependencies, and tools up to date.
- ✓ Triage and fix alerts regularly.
- ✓ Use policies and status checks to enforce security.
- ✓ Educate your team and make security everyone's responsibility.
- ✓ Continuously improve and expand coverage over time.

### KEY TAKEAWAYS



- ✓ Detect and prevent risks before they become problems.
- ✓ Automate security with minimal effort.
- ✓ Give your team actionable insights in the tools they already use.
- ✓ Build a more secure and resilient software supply chain.

Secure earlier. Review faster. Ship safer.



Stay consistent, practice hands-on, and think like a DevOps engineer.  
Build. Test. Learn. Repeat.



You've got this!

