OpenText™ Fortify Code Security Extension for Visual Studio Code

User Guide

Version : 26.2

© Copyright 2026 Open Text

Table of Contents

1. User Guide

Software Version: 26.2

Document Release Date: April 2026

Software Release Date: April 2026

1.1. Change Log

The following table lists changes made to this document. Revisions to this document are published between software releases only if the changes made affect product functionality.

Software Release / Document Version	Changes
26.2	Initial version and release

1.2. Introduction

The Fortify Code Security Extension for Visual Studio Code provides an integrated application security workflow inside Visual Studio Code to support early identification and remediation of vulnerabilities. Use the Fortify Code Security Extension for Visual Studio Code to authenticate, select target applications, run scans, and remediate findings across OpenText Core Application Security and OpenText Application Security Center.

The extension enables local SAST scans using OpenText SAST, submission of remote scans through ScanCentral SAST, and project uploads to Core Application Security for both SAST and Software Composition Analysis (SCA), all from within VS Code.

Security findings are displayed directly in the editor and linked to their corresponding source-code locations for efficient review. The extension also integrates OpenText SAST Aviator to provide AI-assisted remediation, including contextual explanations and one-click code fixes for supported vulnerabilities. By embedding these capabilities into the development workflow, the extension reduces context switching and streamlines security remediation during coding.

With this extension, you can:

- Dynamic command UI generation based on preferred platform (OpenText Core Application Security or Application Security Center).
- Platform-aware workflow orchestration for OpenText Core Application Security and Application Security Center.
- Authenticate to OpenText Core Application Security or Application Security Center from the Visual Studio Code sidebar.
- Select application version/release to scan and remediate.
- Run scans locally with SAST, translate remotely with ScanCentral SAST, or retrieve existing issues.
- Review and audit vulnerabilities in the remediation panel.
- Apply AI-assisted remediations using Fortify Remediation Aviator auto-remediation.

Advanced features

- Interact with the extension using fcli commands from Command Palette.
- Unified scan and remediation flow with Quick Links
- Save configurations for repeated tasks in Favorites.
- Sidebar actions for authentication, application selection, start scan, remediation, and auto-remediation (for Application Security Center workflows only).

- VS Code GitHub Copilot assistance to perform tasks across both OpenText Core Application Security and Application Security Center. Use Copilot Chat to query vulnerabilities via Skills and MCP servers.

1.2.1. Installing the Fortify Code Security Extension for Visual Studio Code

You can install the extension on Windows, Linux, or macOS.

To install the extension, search `Fortify Code Security` in the Visual Studio Marketplace.

For more information and instructions about how to install an extension, see the *Visual Studio Code Documentation*.

After you install the Fortify Code Security Extension for Visual Studio Code, the Visual Studio Code Activity Bar now includes the Fortify Code Security menu.

1.2.2. Software requirements

This topic describes the OpenText Application Security Software that the work with and the requirements for each task.

Before you use the extension, make sure the following requirements are met.

Software	Version	Requirements
Core Application Security (Fortify on Demand)	N/A	To upload your project to Core Application Security for assessment, make sure that you have the following: • ScanCentral SAST standalone client installed and configured explicitly in the extension settings. • Core Application Security credentials and personal access tokens.
OpenText SAST (Fortify Static Code Analyzer)	24.2.0 or later	To scan your project locally with Fortify Static Code Analyzer, you must have the full path to the sourceanalyzer executable. Make sure that your system meets the requirements for the Fortify Static Code Analyzer version you are using as described in the *Fortify Static Code Analyzer User Guide* in Static Application Security Testing Documentation.

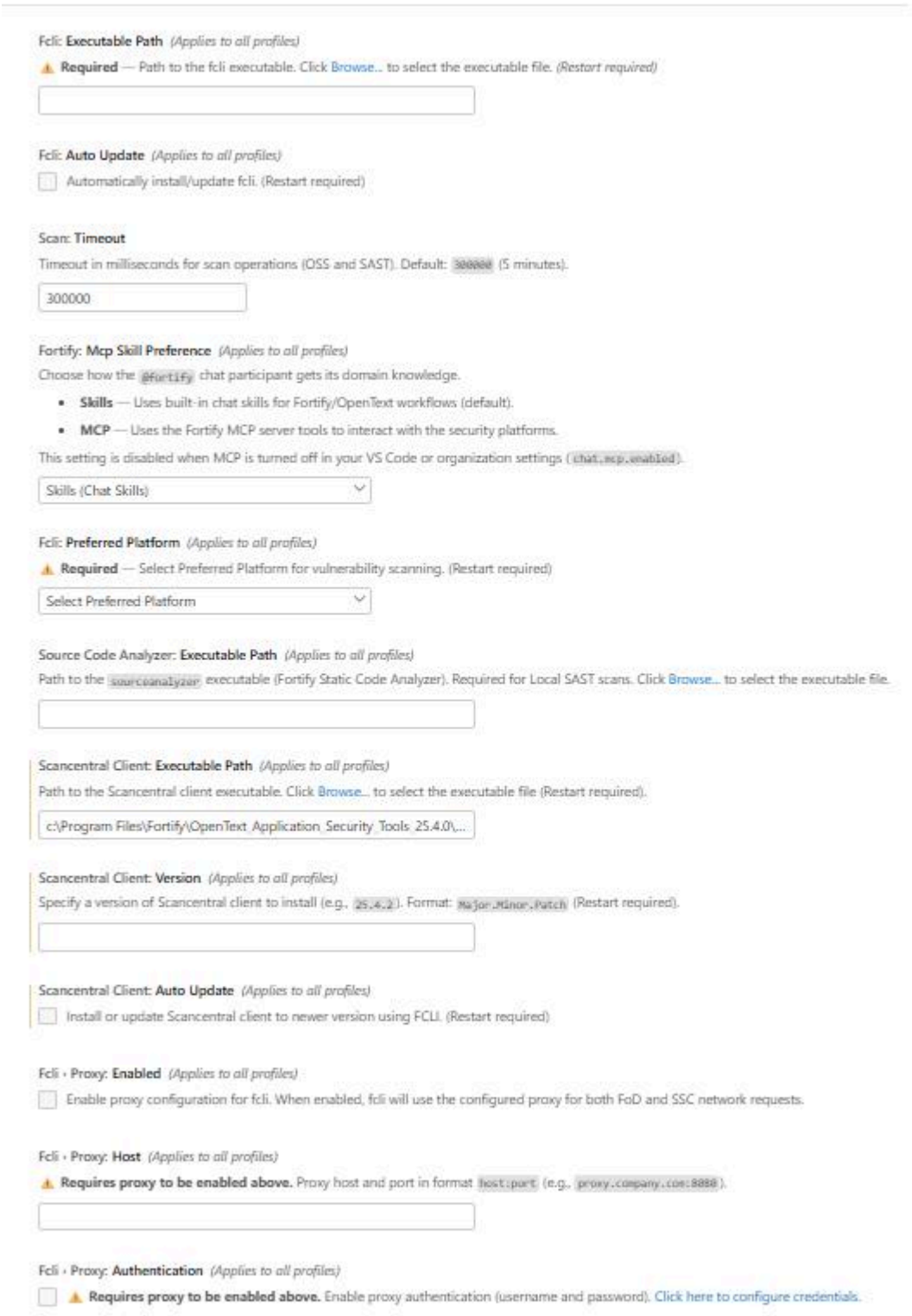
Software	Version	Requirements
OpenText Application Security Center (Software Security Center)	24.2.0 or later	To upload analysis results to OpenText Application Security Center after an analysis with ScanCentral SAST, make sure you have the following: • An application version that exists in OpenText Application Security Center • An authentication token of type `ToolsConnectToken` or `CIToken`
ScanCentral SAST	24.2.0 or later	To scan your project remotely with ScanCentral SAST, make sure that you have a ScanCentral SAST client authentication token. For languages that are supported for analysis and system requirements for the ScanCentral SAST version you are using, see the *Fortify Software System Requirements* document in Application Security Center Documentation.

- Install Visual Studio Code 1.106.3 or later.
- Install Fortify Code Security Extension for Visual Studio Code from the Visual Studio Marketplace.
- Install fcli (the extension automatically installs or updates fcli if **Fcli: Auto Update** is selected in the extension settings) and ensure fcli executable path is configured explicitly in the extension settings.
- Ensure you have access to Core Application Security (Fortify on Demand) or Application Security Center and the required credentials or tokens.
- For local SAST scans, install OpenText SAST locally.
- For remote scans, install ScanCentral client locally or use the **ScanCentral client: Auto Update** settings to install or update the ScanCentral client.
- For VS Code Copilot and MCP workflows, ensure GitHub Copilot Chat is available in Visual Studio Code.
- To audit issues in the analysis results, your user account must have audit permissions.
- To add comments to issues or assign custom tags that require comments, your user account must have the permission to comment on issues.

1.2.3. Configuring the Fortify Code Security Extension for Visual Studio Code

To configure the Fortify Code Security Extension for Visual Studio Code extension settings:

1. Open VS Code **Settings** and search for `Fortify Code Security`. Alternatively, click **Fortify Code Security** in the activity bar and click **SETTINGS** in the sidebar.



2. Configure the following properties for Fortify Code Security Extension:

Property	Description
Fcli: Executable Path	Required. Path to the fcli executable. Click **Browse...** to select the executable file.  **Example** *<home directorv>/fcli-windows/fcli.exe* Restart VS Code to apply the changes.
Fcli: Auto Update	Optional. Select this property to automatically install pr update the fcli version. Restart VS Code to apply the changes.
Fcli: Preferred Platform	Required. Specifies the preferred platform. You can either select Application Security Center (SSC) or Core Application Security (FoD). Restart VS Code to apply the changes.
Scan: Timeout	Optional. Specifies the timeout for any operations in milliseconds. Default: `300000` (5 minutes)

Property	Description
Fortify: Mcp Skill Preference	Specifies how the GitHub Copilot retrieves domain knowledge when you interact with GitHub Copilot Chat. ◦ **Skills**: Uses built-in chat skills for fcli, Core Application Security (Fortify on Demand), and OpenText Application Security Center (Software Security Center) operations. ◦ **MCP**: Uses the Model Context Protocol (MCP) server tools to directly interact with the security platforms through fcli. This option provides real-time access to platform data and commands. **Note** This setting is disabled when MCP is turned off globally in Visual Studio Code settings or by your organization's policies.
Static Code Analyzer: Executable Path	Required for local SAST scans. Specifies the path to the locally installed OpenText SAST executable (`sourceanalyzer.exe`).

Property	Description
ScanCentral Client: Executable Path	Specifies the file path to ScanCentral SAST client executable. Go to the ScanCentral SAST installation directory and do one of the following: ◦ If you are using a standalone client installed with OpenText™ Application Security Tools, navigate to `<tools_install_dir>/bin/` and select `scancentral.bat` (on Windows) or `scancentral` (on non-Windows). ◦ If the standalone client is installed in a different location, navigate to the installation directory and select `scancentral.bat` (on Windows) or `scancentral` (on non-Windows). Restart VS Code to apply the changes. **Note** You cannot configure both **ScanCentral Client: Executable Path** and **ScanCentral Client: Version** at the same time. Select one option to specify or install the ScanCentral client (`scancentral.bat` or `scancentral`).

Property	Description
ScanCentral Client : Version	Optional. Specifes a version of ScanCentral client to install. Use the following version format: `Major.Minor.Patch` Example: 26.2.0 Restart VS Code to apply the changes. **Note** You cannot configure both **ScanCentral Client: Executable Path** and **ScanCentral Client: Version** at the same time. Select one option to specify or install the ScanCentral client (`scancentral.bat` or `scancentral`).
ScanCentral Client: Auto Update	Optional. Enables automatic ScanCentral client install or update through fcli.
Fcli › Proxy: Enabled	(Optional) Select the check box if your environment requires a proxy server to connect to Application Security Center (SSC) or Core Application Security (FoD).
Fcli › Proxy: Host	(Required if **Proxy: Enabled** is selected). Type the host name and port number of the proxy server. Use the following format: `host:port` Example: `proxy.company.com:8080`

Property	Description
Fcli › Proxy: Authentication	(Required if **Proxy: Enabled** is selected). Select the check box if the proxy server requires authentication. Click the **Click here to configure credentials** button. When prompted, type the proxy username and proxy password for the proxy server.

1.2.4. Related Documents

This topic describes documents that provide information about OpenText™ Application Security software products.

All Products

The following documents provide general information for all products. Unless otherwise noted, these documents are available on the Product Documentation website for each product.

Document / File Name	Description
About Fortify Software Documentation appsec-docs-n-*<version>*.pdf	This paper provides information about how to access OpenText Application Security Software product documentation. **Note** This document is included only with the product download.
Fortify Software System Requirements appsec-sr-*<version>*.pdf	This document provides the details about the environments and products supported for this version of OpenText™ Application Security Software.
OpenText™ Application Security *Software Release Notes* appsec-rn-*<version>*.pdf	This document provides an overview of the changes made to OpenText™ Application Security Software for this release and important information not included elsewhere in the product documentation.
What's New in OpenText™ Application Security *Software <version>* appsec-wn-*<version>*.pdf	This document describes the new features in OpenText™ Application Security Software products.

OpenText™ ScanCentral SAST

The following document provides information about OpenText™ ScanCentral SAST. Unless otherwise noted, this document is available on the Product Documentation website at https://www.microfocus.com/documentation/fortify-software-security-center.

Document / File Name	Description
Fortify ScanCentral SAST Installation, Configuration, and Usage Guide sc-sast-ugd-*<version>*.pdf	This document provides information about how to install, configure, and use ScanCentral SAST to streamline the static code analysis process. It is written for anyone who intends to install, configure, or use ScanCentral SAST to offload the resource-intensive translation and scanning phases of their OpenText SAST process.

OpenText™ Application Security

The following document provides information about OpenText™ Application Security. Unless otherwise noted, this document is available on the Product Documentation website at https://www.microfocus.com/documentation/fortify-software-security-center.

Document / File Name	Description
OpenText Application Security (Fortify Software Security Center) User Guide ssc-ugd-*<version>*.pdf	This document provides OpenText™ Application Security users with detailed information about how to deploy and use OpenText Application Security Center. It provides all of the information you need to acquire, install, configure, and use OpenText Application Security Center. It is intended for use by system and instance administrators, database administrators (DBAs), enterprise security leads, development team managers, and developers. OpenText Application Security Center provides security team leads with a high-level overview of the history and current status of a project.

OpenText SAST

The following documents provide information about OpenText SAST. Unless otherwise noted, these documents are available on the Product Documentation website at https://www.microfocus.com/documentation/fortify-static-code-analyzer-and-tools.

Document / File Name	Description
Fortify Static Code Analyzer User Guide sast-ugd-*<version>*.pdf	This document describes how to install and use Fortify Static Code Analyzer to scan code on many of the major programming platforms. It is intended for people responsible for security audits and secure coding.
Fortify Static Code Analyzer Custom Rules Guide sast-cr-ugd-*<version>*.zip	This document provides the information that you need to create custom rules for Fortify Static Code Analyzer. This guide includes examples that apply rule-writing concepts to real-world security issues. **Note** This document is included only with the product download.
Fortify License and Infrastructure Manager Installation and Usage Guide lim-ugd-*<version>*.pdf	This document describes how to install, configure, and use the Fortify License and Infrastructure Manager (LIM), which is available for installation on a local Windows server and as a container image on the Docker platform.

Fortify Static Code Analyzer Applications and Tools

The following documents provide information about Fortify Static Code Analyzer applications and tools. Unless otherwise noted, these documents are available on the Product Documentation website at https://www.microfocus.com/documentation/fortify-static-code-analyzer-and-tools.

Document / File Name	Description
OpenText™ Application Security Tools Guide sast-tgd-<*version*>.pdf	This document describes how to install OpenText™ Application Security Tools. It provides an overview of the applications and command-line tools that enable you to scan your code with OpenText SAST, review analysis results, work with analysis results files, and more.
Fortify Audit Workbench User Guide awb-ugd-<*version*>.pdf	This document describes how to use Fortify Audit Workbench to scan software projects and audit analysis results. This guide also includes how to integrate with bug trackers, produce reports, and perform collaborative auditing.
Fortify Plugin for Eclipse User Guide ep-udg-<*version*>.pdf	This document provides information about how to install and use the Fortify Complete Plugin for Eclipse.
Fortify Analysis Plugin for IntelliJ IDEA and Android Studio User Guide iap-udg-<*version*>.pdf	This document describes how to install and use Fortify Analysis Plugin for IntelliJ IDEA and Android Studio.
Fortify Extension for Visual Studio User Guide vse-ugd-<*version*>.pdf	This document provides information about how to install and use the Fortify extension for Visual Studio to analyze, audit, and remediate your code to resolve security-related issues in solutions and projects.

1.3. Using Fortify Code Security Extension with Core Application Security

You must have the standalone ScanCentral SAST client on the system where Fortify Code Security Extension for Visual Studio Code is installed to upload code to OpenText Core Application Security.

Requirements

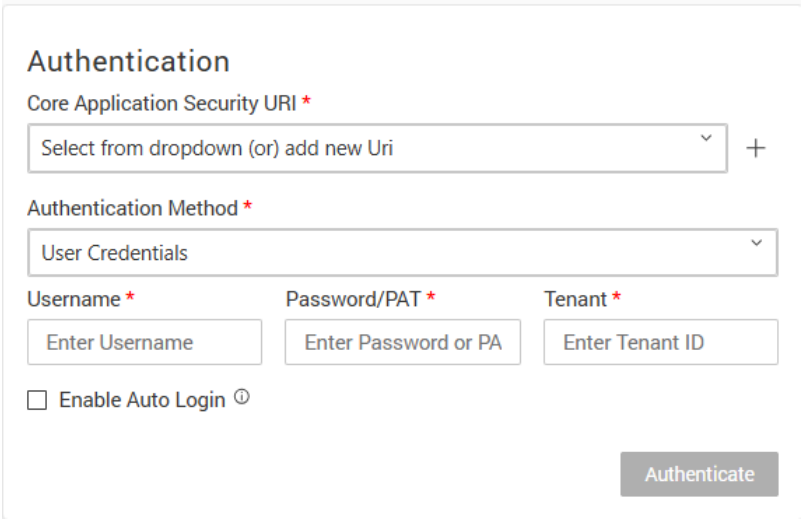
- fIci must be installed (the extension automatically installs or updates fcli if **Fcli: Auto Update** is selected in the extension settings) and configured with **Fcli: Executable Path** property in the VS Code settings.
- The **Preferred Platform** in the VS Code settings must be set to **Core Applicatin Security (FoD)**.
- The Core Application Security tenant must have the required permissions to view applications, run scans, and update issue audit fields.

To upload the opened project to OpenText Core Application Security for assessment:

Authenticate your OpenText Core Application Security account

1. If the extension is not open, click **Fortify Code Security** in the activity bar.
2. Click **AUTHENTICATE** in the sidebar.

   The Authentication box is displayed in the VS Code Editor.



3. From the **Core Application Security URI** list, you can either select one of the available URIs.

   Alternatively, if the URI is not available:

   1. Click + to add a new URI to the **API Root URI** list.

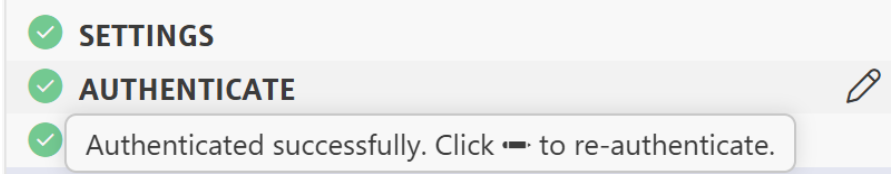
2. Type a valid URI.
3. Click ✓. If the URI is valid, you can view the new URI in the **Core Application Security URI** list.

4. Select an authentication method and provide the relevant credentials described in the following table.

Authentication method	Procedure
Username Credentials	1. In the **Username** box, type the account username. 2. In the **Password/PAT** box, type the account password or the personal access token. 3. In the **Tenant** box, type the tenant ID.
Client Credentials	1. In the **Client Id** box, type the API key. 2. In the **Client Secret** box, type the API secret.
Enable Auto Login	(Optional)Select the check box to securely store credentials and automatically restore expired sessions or silently reauthenticate when a session is missing or expired. No login prompts are displayed.

5. Click **Authenticate**.

   Upon successful authentication, **Authenticated** status is displayed. Otherwise, the status displays **Not Authenticated**.

6. 

To modify the API root URL, authentication method, or credentials, click **Reauthenticate**.

Select Application and Release

1. Click **APPLICATION** in the sidebar.
2. In the **Select Application** area, select an application from the **Application Name** list.

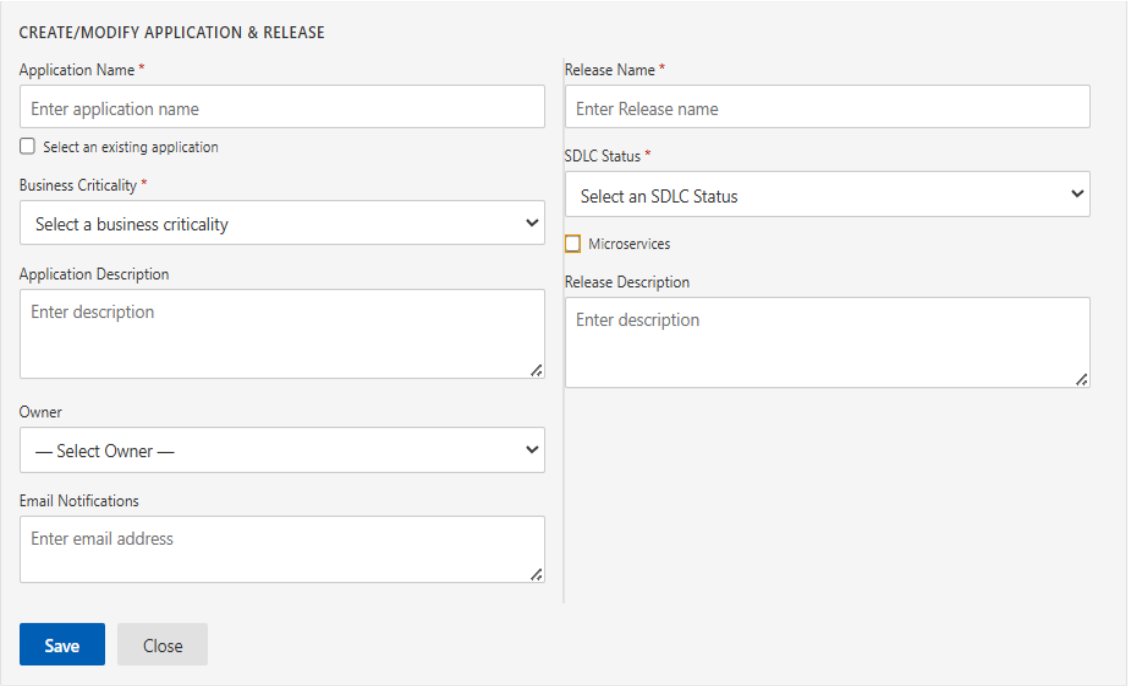
   > **Note**
   > Only 50 entries are displayed in the **Application Name** list.
   > When you search for an application, the extension searches only within these initial 50 entries.
   > To view and search additional entries, click **Load next 50 applications...** to load the next set of 50, and repeat as needed.

3. Select a release from the **Release** list.

   > **Note**
   > Only 50 entries are displayed in the **Version** list. To view additional entries, click **Load next 50 versions...** to load the next set of 50, and repeat as needed.

4. Alternatively, to create a new Application or Release or modify an existing application or release:
   1. click **Create/Modify Application**.

      The **CREATE/MODIFY APPLICATION & RELEASE** panel displays in the VS Code editor.



2. Define the application. Fields are required, unless otherwise noted.

Property	Description
Application Name	Type the name of your application or use the **Select an existing application** check box to modify an existing application.
Select an existing application	(Optional) To modify an existing application, select the check box. Select or search for an application from the **Application Name** list. When you select an application, the application attribute values are synced from Core Application Security and populated in the respective fields.
Business Criticality	Select the application's level of importance: ▪ **High**: Security issues could have catastrophic consequences for the business. ▪ **Medium**: Security issues would have non-trivial consequences, but ones which do not pose a life-or-death threat to the business. ▪ **Low**: Security issues can be ignored or addressed gradually as time permits

Microservices	(**Web / Thick-Client** applications only) Select the check box to scan the application as a microservice application. Type the name of a microservice in the **Microservice name** box and click +**Add.** The microservice is added. You can add up to 10 microservices. **Important** The designation of a microservice application is permanent and cannot be changed after the application has been created.
Application Description	(Optional) Type a description of the application that will help you manage multiple applications.
Owner	(Optional) Owner of the release who receives email notifications of scan status updates to the release
Email Notifications	(Optional) List the email addresses that will receive email notifications of scan status updates for the application. Separate multiple email addresses with a semicolon or comma.

3. After you specify the application details, you can define the release. Fields are enabled after you specify the application details. Fields are required, unless otherwise noted.

Property	Description
Release Name	Type the name of your release.
Select an existing Release	If you have selected the **Select an existing application** check box, you can use the **Select an existing Release** check box to modify an existing release. Select or search for a release from the **Release Name** list. When you select a release, the application release attribute values are synced from Core Application Security and populated in the respective fields.
SDLC Status	Select the Software Development Life Cycle stage of the release: **Development**, **QA/Test**, **Production**. The **Retired** option is not available.
Microservice (microservice applications only)	Select the microservice that will be linked to the release from the drop down list. A release must be linked to a microservice.
Release Description	(Optional) Type a description that helps describe the release.

4. Click **Save**. The application name and release is displayed in the sidebar.
5. Click **Select Application.**

The application name and release is displayed in the sidebar.

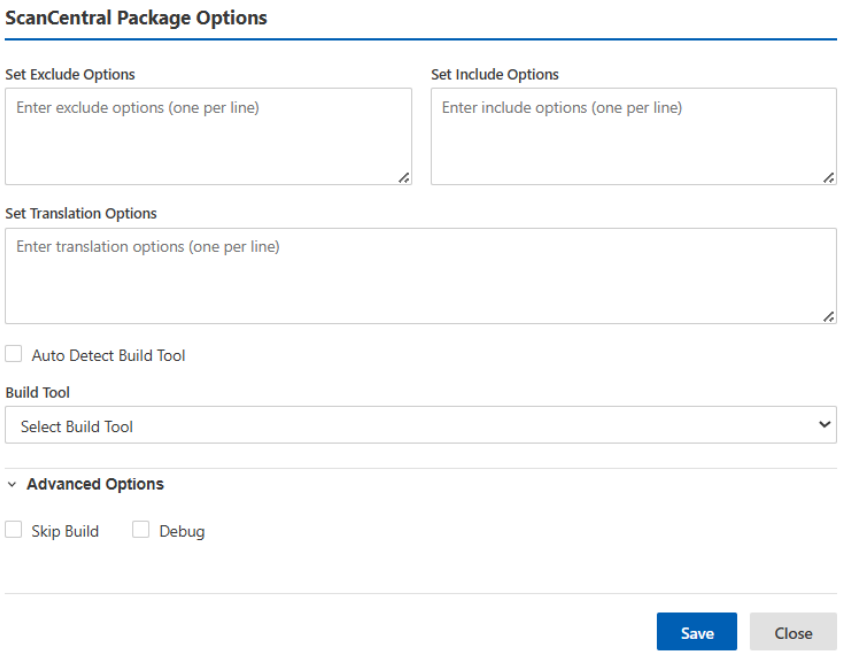
Note

You can add and modify custom attributes for application or releases that are marked Required for the OpenText Core Application Security tenant.

Scanning the Application

1. Click **START SCAN** in the sidebar.
2. In the **PACKAGE OPTIONS** area, click **View/Edit**.

   The ScanCentral Package Options box is displayed.



3. Provide the options described in the following table. All field are optional, unless otherwise noted.

Option	Description
Set Exclude Options	Specify the relative paths of files or directories to exclude from the package separated by a new line (one per line). You can use wildcard characters to specify the paths.
Set Include Options	Specify the relative paths of files or directories to include in the package separated by a new line (one per line). You can use wildcard characters to specify the paths. If you leave it empty, all the supported files are included.
Set Translation Options	Specify a list of OpenText SAST translation options separated by a new line (one per line).

Option	Description
Auto Detect Build Tool	Select this option to automatically detect the build tool. **Note** For .NET projects: ◦ On Windows, if you select the **Auto Detect build tool** option, the default build tool is MSBuild. ◦ On Linux, if you select the **Auto Detect build tool** option, the default build tool is DotNet. ◦ If you want to use DotNet as the build tool on a Windows, you must clear the **Auto Detect build tool** option and explicitly select **DotNet** from the **Build Tool** list. Selecting the **Auto Detect Build Tool** check box automatically hides the **Build Tool** list and **Advanced Options**.
Skip Build	Select whether to skip the build invocation that prepares the generated sources and libraries before the project information is packaged for submission to ScanCentral SAST.
Set Debug	Select this option to enable debug logging that can be helpful to troubleshoot issues.

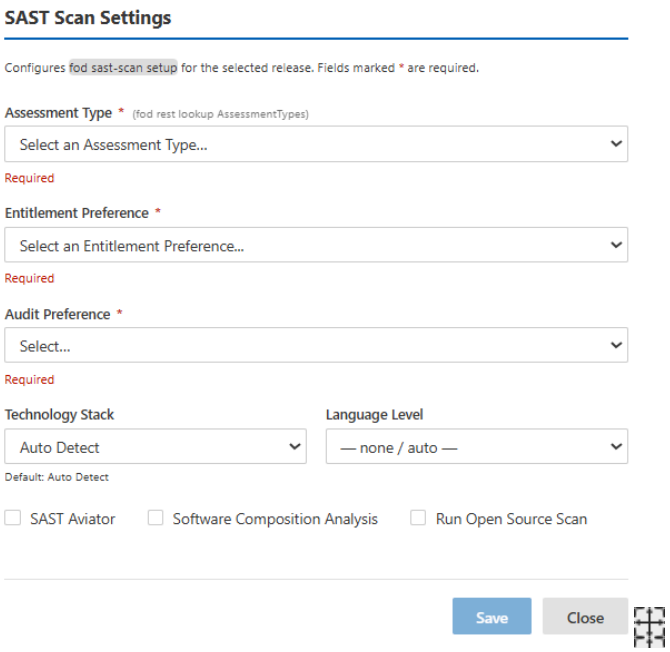
4. To select a build tool explicitly, clear the **Auto Detect Build Tool** check box and select a build tool in the **Build Tool** list and provide the settings based on the build tool.

Option	Description
Build File	(Optional) (For Gradle or Maven) Click **Browse** and select the build file if it is different than the default of `build.gradle` or `pom.xml`. (For DotNet or MSBuild) Type the name of the build file. If you do not select a build file, ScanCentral SAST automatically detects the build file.
Build Command	(Optional) Type any custom build commands to prepare and build the project. If not specified, the default build command is used.
Advanced Options	
PHP Version	(Optional) Type the PHP version used in the project.
Python Virtual Environment Location	(Optional) Click **Browse** and select the location (directory) of the Python virtual environment. Specify this together with the Python requirements file to have dependencies restored before the scan.

Option	Description
Python Requirements File	(Optional) Click **Browse** and select the Python project requirements file used to install and collect dependencies. Use only this Python field if you have no preference for the Python version used or there is only one Python version installed and on the PATH.
Python Version	(Optional) Select the Python version for Python projects.

5. Click **Save**.
6. In the **Scan Settings** area, you can select either a Static assessment or an open source software composition analysis in conjunction with a static assessment or as a separate open source only assessment.
    1. In the **Scan Settings** area, select **Static + Open Source.**
        1. If the status displays **Not Configured**, click **View/Edit**.

           The Scan Settings box displays.



        2. Complete the fields as needed. All field are required, unless otherwise noted.

Field	Description
Assessment Type	Select **Static+ Assessment** or **Static Assessment**.
Entitlement Preference	Select the entitlement that the assessment will use. The field displays entitlements that are valid for the selected assessment type, including those available for purchase. Note that microservice applications are restricted to subscriptions. If the release has an active subscription, only options that do not consume entitlements are displayed.

Field	Description
Audit Preference	Select the audit preference. ▪ **Manual**: False positives identified by Fortify Audit Assistant with high confidence are automatically suppressed. A security expert then manually reviews the scan results. ▪ **Automated**: False positives identified by Fortify Audit Assistant with high confidence are automatically suppressed and results are published without manual review. **Note** If you select the SAST Aviator check box, Automated audit is selected by default. If you change the **Audit Preference** to **Manual**, the **SAST Aviator** check box is cleared automatically. The ability to select audit preference depends on the assessment type: ▪ A Static single scan allows Automated only.

Field	Description
	▪ A Static subscription allows one Manual audit per application (not per release or microservice). ▪ A Static+ single scan allows Manual only. ▪ A Static+ subscription allows Automated or Manual audit for each assessment.
Technology Stack	Select the application's technology stack. The languages available for selection depends on the application type and whether the application is a microservice application.
Language Level	If applicable, select the technology stack's language level from the list.

Field	Description
Include Third Party Libraries	(Optional) Select the check box to have third party libraries scanned for vulnerabilities, which will be included in the scan results. This significantly increases the turnaround time. This option is not available for microservice applications. **Note** Selecting this option infers that your organization has received consent from all third-party vendors to scan their libraries.
SAST Aviator	(Optional) For scans using Automated audit, select the check box to have SAST Aviator audit results and provide enhanced remediation assistance.
Run Open Source Scan	(Optional) Select the check box to include open source software composition analysis. No code leaves the OpenText Core Application Security environment.

3. Click **Save**.
4. Click **Start Scan** to upload a ScanCentral package to OpenText Core Application Security.

2. In the **Scan Settings** area, Select **Open Source Only**.
    1. If the status displays **File Not Selected**, click **Upload a File**.

       The status displays the zip file name or the Current Project.
    2. Click **Start Scan** to upload the package to OpenText Core Application Security.
7. If the project is successfully uploaded, the Fortify Code Security Extension for Visual Studio Code displays the completion status and the scan ID in the **OUTPUT** console in `FCli JSON-RPC` channel.

   If you want to reset the saved options, click **Reset**.

Remediating your code

After you select an application release, you can view the vulnerabilties and remediate.

Viewing and selecting vulnerabilities

To view and select vulnerabilities in the selected application release:

1. Click **REMEDIATE** in the sidebar.

   From the **GroupBy** list, select an attribute for sorting vulnerabilities in all visible folders into groups.

   The default grouping is **Category**.
2. Click a tab to view the associated vulnerabilities.

   Note
   The tabs shown depend on the issue severity of the application release.

   - The **Critical** tab contains issues that have a high impact and a high likelihood of exploitation. We recommend that you remediate critical issues immediately.
   - The **High** tab contains issues that have a high impact and a low likelihood of exploitation. We recommend that you remediate high issues with the next patch release.
   - The **Medium** tab contains issues that a have low impact and a high likelihood of exploitation. We recommend that you remediate medium issues as time permits.
   - The **Low** tab contains issues that have a low impact and a low likelihood of exploitation. We recommend that you remediate low issues as time permits

(your organization can customize this category).

- The **All** tab contains all issues.

Within each tab, issues are grouped by vulnerability category. After each grouping name, enclosed in brackets, is the number of audited issues and the total number of issues in the group.

3. To view fixed and/or suppressed issues, click **Fixed** and/or **Suppressed** from the **Visibility** list.
4. Click to expand a grouping and view the issues it contains.

   The Fortify Code Security Extension for Visual Studio Code retrieves the corresponding issues from OpenText Core Application Security.

5. Select an issue to view its details in the **ISSUE AUDIT** panel.
6. Click the **Refresh** icon to sync any changes to the issues in the application release in OpenText Core Application Security.

Grouping vulnerabilities

The following grouping options are available on the **REMEDIATE** section:

Group	Description
Assigned User	User defined
Auditor Status	Auditor issue remediation status.
Category	Issue category
CWE Top 25 2023	Common Weakness Enumeration Top 25 2023 classification
CWE Top 25 2024	Common Weakness Enumeration Top 25 2024 classification
CWE Top 25 2025	Common Weakness Enumeration Top 25 2025 classification
Developer Status	Developer issue remediation status
FISMA	FISMA classification (deprecated)
Fortify Aviator	Audited by Fortify Remediation Aviator
Attachments	Has attachments
Comments	Has Comments
NIST SP 800-53 Rev. 5	National Institute of Standards and Technology Special Publication 800-53
OWASP 2014 Mobile	OWASP mobile top 10 2014 classification
OWASP 2017	OWASP top 10 2017 classification
OWASP 2021	OWASP top 10 2021 classification
OWASP 2025	OWASP top 10 2025 classification

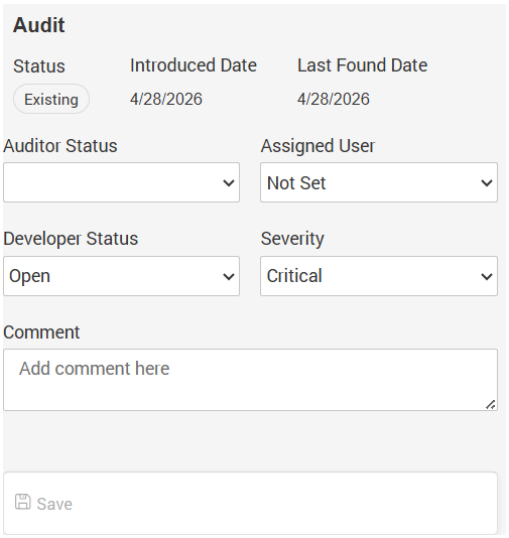
Group	Description
OWASP LLM Top 10 2025	OWASP LLM top 10 2025 classification
OWASP ASVS 4.0	OWASP ASVS 4.0 classification
OWASP ASVS 5.0	OWASP ASVS 5.0 classification
OWASP Mobile Top 10 2024	OWASP mobile top 10 2024 classification
Package	Package or namespace
PCI 4.0	PCI 4.0 classification
PCI 2.0	PCI 2.0 classification
PCI 3.0	PCI 3.0 classification
PCI 3.1	PCI 3.1 classification
PCI 3.2	PCI 3.2 classification
PCI SSF 1.2	PCI SSF 1.2 classification
Scan Type	Scan type to which issue belongs
Severity	Issue severity (filter only)
Sink	Dataflow sink function, applicable for static scan issues
Source	Dataflow source function, applicable for static scan issues
Status	Issue status
<Custom attribute>	User defined custom attributes

Viewing issue information

After you select an issue in the **ISSUE AUDIT** panel, the Fortify Code Security Extension for Visual Studio Code displays the issue-specific content in the in the **ISSUE AUDIT** panel on the **Audit**, **Analysis Trace**, **Vulnerablities**, **Recommendations**, and **History** tabs.

Auditing issues

The **AUDIT** view provides a dashboard of analysis information for the selected issue.



If you have the Edit Issues permission, you can assign a user, set the developer status, and add comments for issues in the Audit Summary view. If you have the Audit Issues permission, you can also edit the issue's auditor status and severity.
To audit an issue:

1. Make sure that the **Audit** panel is open.
2. From the issues list in the **REMEDIATE** section, select an issue. You can select multiple issues to make the same edits to multiple issues.
3. In the **Audit** panel, select the user to assign to the issue from **Assigned User** list.
4. To change the issue's development status, select the status from the **Developer Status** list
5. To change the auditor status, select the status from the **Auditor Status** list.
6. To change the issue severity, select an issue severity from the **Severity** list.
7. To add a comment for the issue, type your comment in the box at the bottom of the **Comments** area..
8. Click **Save**.

   The changes and comments are displayed in the **History** tab.

The Fortify Code Security Extension for Visual Studio Code saves your changes for the selected application release.

Note

Any changes you make on the **Audit** tab are automatically uploaded to the application release in Core Application Security.

Analysis Trace

The **Analysis Trace** tab displays the relevant trace output. This is a set of program points that show how the analyzer found the issue. VS Code places the focus on the line of code that contains the selected security-related issue.

For dataflow issues, this trace is a presentation of the path that the tainted data follows from the source function to the sink function. For example, when you select an issue that is related to potentially tainted dataflow, the analysis trace box shows the direction of the dataflow in this section of the source code.

Analysis Trace Icons

The analysis trace icons described in the following table show how dataflow moves in the section of the source code or execution order.

Icon	Description	Icon	Description
	Data is assigned to a field or variable		Tainted data is returned from a function
	Information is read from a source external to the code such as an HTML form or a URL		A pointer is created
	Data is assigned to a globally scoped field or variable		A pointer is dereferenced
	A comparison is made		The scope of a variable ends
	The function call receives tainted data		The execution jumps
	The function call returns tainted data		A branch is taken in the code execution

Icon	Description	Icon	Description
	Passthrough, tainted data passes from one place to another This is typically shown as functionA(x : y) to indicate that data is transferred from x to y. The x and y values are one of the following: • An argument index • `return` —The return value of a function • `this` —The instance of the current object • A specific object field or key		A branch is not taken in the code execution
	An alias is created for a memory location		Generic

Icon	Description	Icon	Description
	Data is read from a variable		A runtime source, sink, or validation step
	Data is read from a global variable		Taint change

Vulnerability

The **Vulnerability** tab displays the following technical details about the issue: issue summary; explanation of the execution and implications of the issue; instance ID and primary rule ID; and standards and best practices information from Fortify Software Security Research.

Recommendations

The **Recommendations** tab displays remediation information and references for further research. The following table describes the sections on this tab.

Section	Description
Recommendations/Custom Recommendations	Describes possible solutions for the selected issue. It can also include examples and recommendations defined by your organization.
Tips/Custom Tips	Provides useful information specific to the selected issue, and any custom tips defined by your organization.
References/Custom References	Lists references for the recommendations provided, including any custom references defined by your organization.

History

The **History** tab displays a log of the following issue events: audit changes, comments, and system events (status changes, copy state actions). You can filter the log by the event type (audit, comment, or system event). You can also refresh the history to fetch the latest events.

Auditing multiple issues

You can select multiple issues in an application release and bulk audit multiple issues.

To bulk audit multiple issues:

1. Select an application release.
2. In **REMEDIATE** section, select the check boxes next to the issues.

   The Fortify Code Security Extension for Visual Studio Code displays the selected issues under the **Selected Issues** area in the **ISSUE AUDIT** tab.

   The **Selected Issues** area displays the total number of selected issues, issue ID, release, and primary location.
3. Click an issue ID under the **Issue ID** column to view the audit information, analysis trace, vulnerability, recommendations, and history for each individual issue.
4. Click the **Multi Selection** button to navigate to the previous page.
5. In the audit panel, edit the fields as needed. The following fields are available for editing: **Assigned User, Developer Status, Auditor, Status, Severity, and Comments.**
6. Click **Save**.

1.4. Using Fortify Code Security Extension with Application Security Center

This section describes the requirements, configuration, and procedure to use OpenText SAST and ScanCentral SAST to analyze and remediate your code.

With the Fortify Security Code Extension, you can either:

- Perform local scan with OpenText SAST.
- Perform the entire analysis (translation and scan) remotely with ScanCentral SAST.
- Perform the translation locally with OpenText SAST and then automatically upload the translated project to ScanCentral SAST for the scan phase.

  Note
  You must have a locally installed and licensed OpenText SAST to perform the translation phase.
  OpenText strongly recommends that you periodically update the security content.

- Review and audit issues from Application Security Center and remediate the issues in VS Code.
- Auto-remediate the issues using Fortify Remediation Aviator recommendations

Requirements

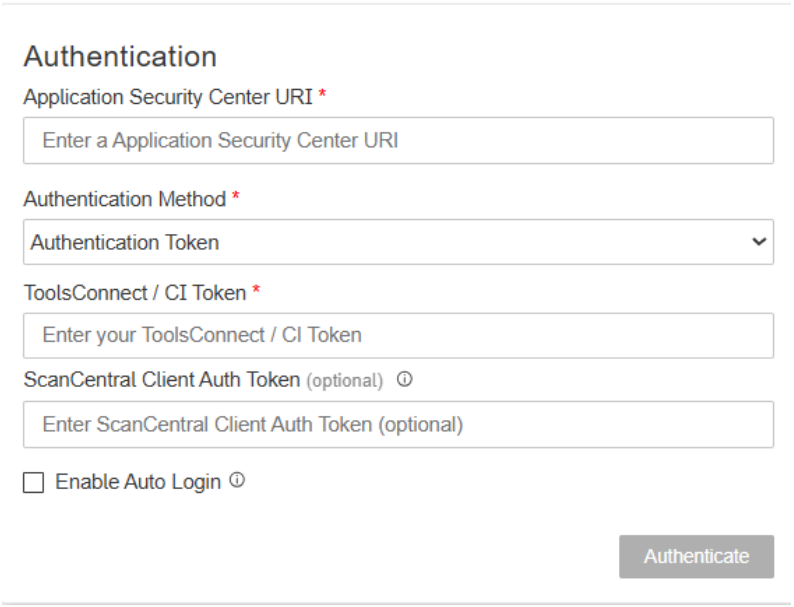
- A locally installed and licensed OpenText SAST with Application Security Content to run local SAST translations and scans.
- A properly configured ScanCentral SAST installation or use the **ScanCentral client: Auto Update** settings to install or update the ScanCentral client. For more information, see the *OpenText™ ScanCentral SAST Installation, Configuration, and Usage Guide* in OpenText Application Security Center Documentation.
- An Application Security Center URL and user credentials or an authentication token of type `ToolsConnectToken` or `CIToken`.
- A ScanCentral client authentication token to run remote translations or scans.
- The **Fcli: Preferred Platform** property must be set to Application Security Center (SSC) in the VS Code settings.

- The **Static Code Analyzer: Executable Path** must be configured for local SAST scans.
- The **Scancentral Client: Executable Path** or **ScanCentral Client : Version** must be configured to perform remote scans with ScanCentral SAST.

Authenticate your Application Security Center account

1. If the extension is not open, click **Fortify Code Security** in the activity bar.
2. Click **AUTHENTICATE** in the sidebar.

   The Authentication box is displayed in the VS Code Editor.



3. In the **Application Security Center URI** box, type the URL for your Application Security Center server.
4. Select an authentication method and provide the relevant credentials described in the following table.

Authentication Method	Procedure
Username & Password	1. In the **Username** box, type the account username. 2. In the **Password** box, type the account password. 3. (For remote scans) In the **ScanCentral Client Auth Token**, type the ScanCentral client authentication token.
Authentication Token	1. In the **ToolsConnect / CI Token** box, type the Application Security authentication token of type `ToolsConnectToken` or `CIToken`. 2. In the **ScanCentral Client Auth Token** box, type the ScanCentral client authentication token.

5. (Optional) Select the **Enable Auto Login** check box to securely store credentials and automatically restore expired sessions or silently reauthenticate when a session is missing or expired. No login prompts are displayed.
6. Click **Authenticate**.

   Upon successful authentication, `Authenticated successfully` status is displayed. Otherwise, the status displays `Not Authenticated`.

   To modify the Application Security Center URL, authentication method, or credentials, click the **Edit** icon to reauthenticate.

Select Application and Version

1. Click **SELECT APPLICATION** in the sidebar.

2. In the **Select Application** area, select an application from the **Application Name** list.

   > **Note**
   > Only 50 entries are displayed in the **Application Name** list.
   > When you search for an application, the extension searches only within these initial 50 entries.
   > To view and search additional entries, click **Load next 50 applications...** to load the next set of 50, and repeat as needed.

3. Select a version from the **Version** list.

   > **Note**
   > Only 50 entries are displayed in the **Version** list. To view additional entries, click **Load next 50 versions...** to load the next set of 50, and repeat as needed.

4. Alternatively, to create a new application or version or modify an existing application or version:
   1. Click + **Create/Modify Application**.

      The **CREATE / MODIFY APPLICATION & VERSION** panel displays in the VS Code editor.

   2. Define the application and version. Fields are required, unless otherwise noted.

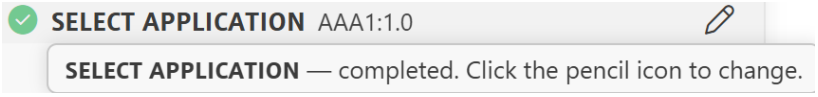
Field	Description
Application Name	Type the application name or use the **Select an existing application** check box to modify an existing application.
Select an existing application	(Optional) To modify an existing application, select the check box. Select or search for an application from the **Application Name** list.
Version Name	Type a name of the version.
Select an existing version	(Optional) If you have selected the **Select an existing application** check box, you can use the **Select an existing version** check box to modify an existing version. Select or search for a release from the **Version Name** list. When you select a version, the application version attribute values are synced from Application Security Center and populated in the respective fields.
Application Version Attributes	
Development Phase	Current phase of development the application version is in.
Development Strategy	Staffing strategy used for application development.

Field	Description
Accessibility	Level of access required to use the application.
Issue Template	Select the check box for a template that sets the minimum thresholds for issue detection. The default template is Prioritized High Risk Issue Template.

3. Click **Save** to save the application version. The application name and version is displayed in the sidebar.

- Click **Select Application.**

  Upon successful completion, the application name and version is displayed in the sidebar. Click **Edit** icon to change the selected application or version.



This section contains the following topics:

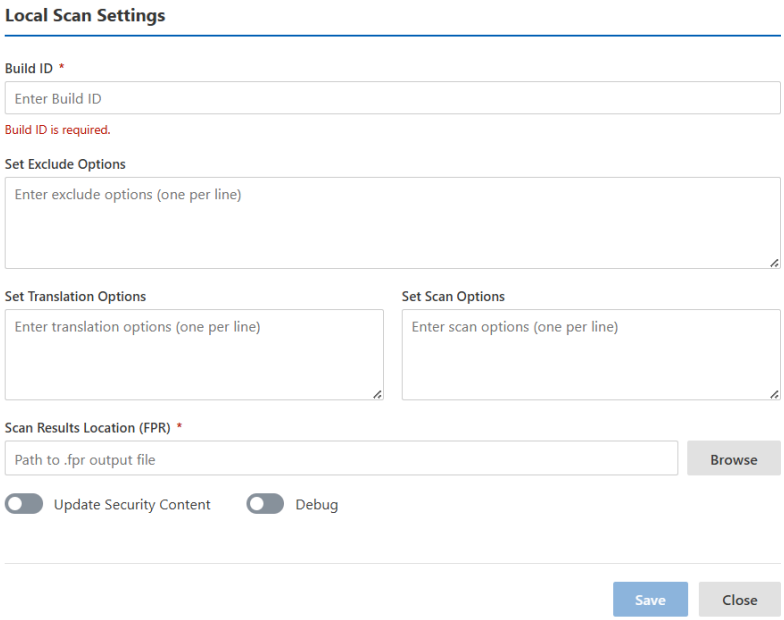
- Performing a local analysis with OpenText SAST
- Performing an analysis remotely with ScanCentral SAST
- Remediate your code in Application Security Center
- Auto-remediate your code in Application Security Center

1.4.1. Performing a local analysis with OpenText SAST

Configuring a local analysis with OpenText SAST

To scan the opened project locally with OpenText SAST:

1. Click **START SCAN** in the sidebar.
2. In the **Scan Type** area, select **Local (SAST)**.
3. In the **LOCAL SCAN OPTIONS** area, click **View/Edit**.



4. Provide the information described in the following table.

Option	Description
Build ID	Required. Type a unique identifier for the analysis.
Set Exclude Options	Optional. Specify the relative paths of files or directories to exclude from the package separated by a new line (one per line). You can use wildcard characters to specify the paths.
Set Translation Options	Optional. Specify a list of OpenText SAST translation options separated by a new line (one per line).
Set Scan Options	Optional. Specify a list of OpenText SAST scan options separated by a new line (one per line).
Scan Results Location (FPR)	Required. Click **Browse** and select the directory where you want to store the scan results file. By default, the extension stores the scan results file in the current workspace.
Update Security Content	Optional. Enable this option to download Fortify security content before the scan.
Debug	Optional. Select this option to display debug information that can be helpful to troubleshoot issues.

5. Click **Save**.
6. (Optional) To upload results to Application Security, turn on the **Upload Local Scan results to Application Security Center** switch.

7. Click **Start Scan**.

   Fortify Code Security Extension for Visual Studio Code automatically detects the OpenText SAST path from the Visual Studio Code extension settings.

   Fortify Code Security Extension for Visual Studio Code starts the scan and displays the status information in the OUTPUT tab in FCli JSON-RPC format. When the scan is complete, Fortify Code Security Extension for Visual Studio Code displays the scan completion status and the Job ID in an information message.

1.4.2. Performing an analysis remotely with ScanCentral SAST

To upload the opened project for analysis by ScanCentral SAST:

1. Click **START SCAN** in the sidebar.
2. In the **Scan Type** area, select **Remote (ScanCentral SAST)**.

Configuring a remote scan with local translation

1. Select **Remote (ScanCentral SAST) > Local Translation** to run the translation phase on the local system and the scan phase with ScanCentral SAST.
2. In the **Local Translation Options** area, click **View/Edit**.

   The **ScanCentral Local Translation Options** dialog box is displayed in the VS Code editor.
3. Complete the fields as needed.

Option	Description
Build ID	Type a unique identifier for the analysis.
Set Exclude Options	Specify the relative paths of files or directories to exclude from the package separated by a new line (one per line). You can use wildcard characters to specify the paths.
Set Translation Options	Specify a list of OpenText SAST translation options separated by a new line (one per line).
Debug	Select this option to set the log level to contain debug information that can be helpful to troubleshoot issues.

4. Click **Save**.
5. (Optional) In the **Remote Scan Options** area, click **View/Edit**.

   The **ScanCentral Remote Scan Options** dialog box is displayed in the VS Code editor.
6. In the **Set Scan Options** field, specify a list of ScanCentral SAST scan options separated by a new line (one per line).
7. Click **Save**.
8. Click **Start Scan**.

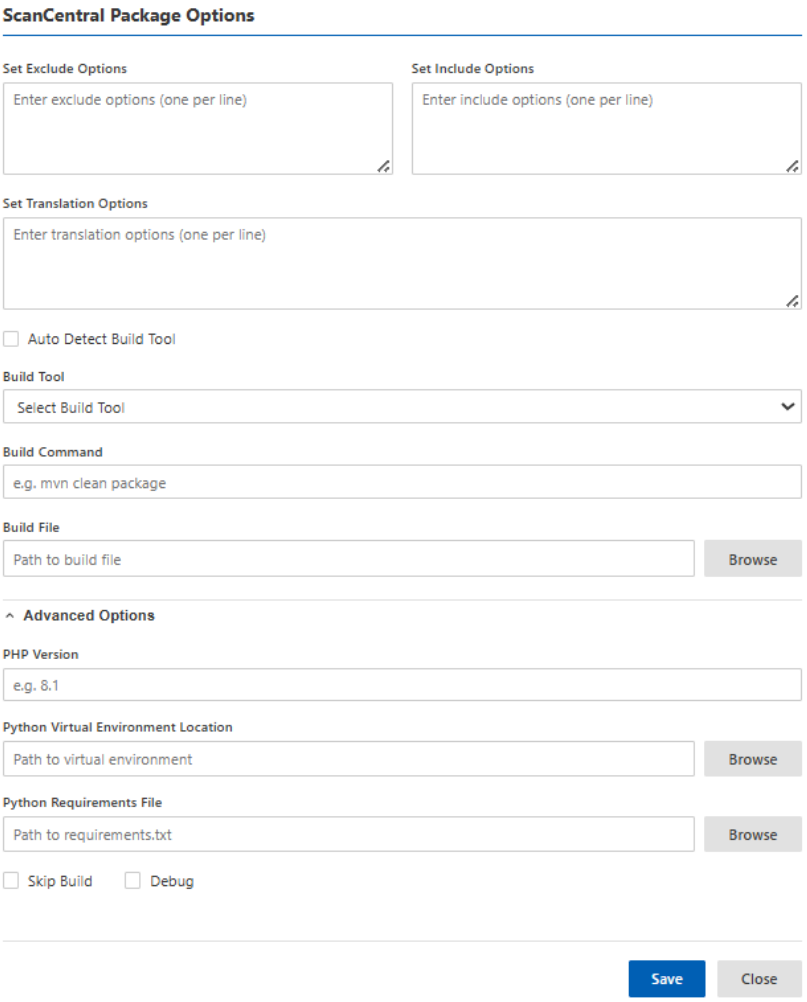
   Fortify Code Security Extension for Visual Studio Code automatically detects the file path to ScanCentral SAST client executable and the locally installed OpenText SAST executable from the Visual Studio Code extension settings.

   Fortify Code Security Extension for Visual Studio Code starts the scan and displays the status information in the OUTPUT console. When the scan is complete, Fortify Code Security Extension for Visual Studio Code displays the scan completion status and the Job ID in an information message.

Configuring a remote translation and scan

1. Select **Remote (ScanCentral SAST) > Remote Translation** to run the translation phase and the scan phase with ScanCentral SAST.
2. In the **Package Options** area, click **View/Edit**.

   The ScanCentral Package Options box is displayed in the VS Code editor.



3. Provide the options described in the following table. All field are optional, unless otherwise noted.

Option	Description
Set Exclude Options	Specify the relative paths of files or directories to exclude from the package separated by a new line (one per line). You can use wildcard characters to specify the paths.
Set Include Options	Specify the relative paths of files or directories to include in the package separated by a new line (one per line). You can use wildcard characters to specify the paths. If you leave it empty, all the supported files are included.
Set Translation Options	Specify a list of OpenText SAST translation options separated by a new line (one per line).

Option	Description
Auto Detect Build Tool	Select this option to automatically detect the build tool. **Note** For .NET projects: ◦ On Windows, if you select the **Auto Detect build tool** option, the default build tool is MSBuild. ◦ On Linux, if you select the **Auto Detect build tool** option, the default build tool is DotNet. ◦ If you want to use DotNet as the build tool on a Windows, you must clear the **Auto Detect build tool** option and explicitly select **DotNet** from the **Build Tool** list.
Skip Build	Select whether to skip the build invocation that prepares the generated sources and libraries before the project information is packaged for submission to ScanCentral SAST.
Set Debug	Select this option to display debug information that can be helpful to troubleshoot issues.

4. To select a build tool explicitly, clear the **Auto Detect build tool** option. The **Build Tool** list is displayed.

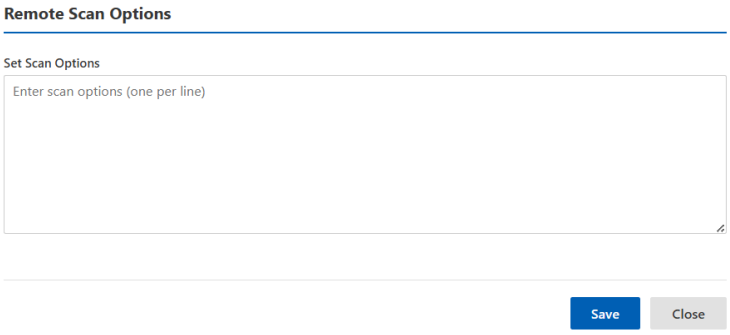
5. Select the build tool in the **Build Tool** list and provide the settings based on the build tool.

Option	Description
Build File	(Optional) (For Gradle or Maven) Click **Browse** and select the build file if it is different than the default of `build.gradle` or `pom.xml`. (For DotNet or MSBuild) Type the name of the build file. If you do not select a build file, ScanCentral SAST automatically detects the build file.
Build Command	(Optional) Type any custom build commands to prepare and build the project. If not specified, the default build command is used.
Advanced Options	
PHP Version	(Optional) Type the PHP version used in the project.
Python Virtual Environment Location	(Optional) Click **Browse** and select the location (directory) of the Python virtual environment. Specify this together with the Python requirements file to have dependencies restored before the scan.

Option	Description
Python Requirements File	(Optional) Click **Browse** and select the Python project requirements file used to install and collect dependencies. Use only this Python field if you have no preference for the Python version used or there is only one Python version installed and on the PATH.
Python Version	(Optional) Select the Python version for Python projects.

6. Click **Save**.
7. (Optional) In the **Remote Scan Options** area, click **View/Edit**.

   The **Remote Scan Options** box is displayed in the VS Code editor.



8. In the **Set Scan Options** field, specify a list of ScanCentral SAST scan options separated by a new line (one per line).
9. Click **Save**.
10. Click **Start Scan**.

    Fortify Code Security Extension for Visual Studio Code automatically detects the file path to ScanCentral SAST client executable from the Visual Studio Code extension settings.

    Fortify Code Security Extension for Visual Studio Code starts the scan and displays the status information in the OUTPUT console. When the scan is complete, Fortify Code Security Extension for Visual Studio Code displays the scan completion status and the Job ID in an information message.

1.4.3. Remediate your code in Application Security Center

After you select an application version, you can view and audit the scan results and remediate issues found in your application.

Viewing and selecting issues

To view and select issues in an opened application **version**:

1. Click **REMEDIATE** in the sidebar.

   From the **GroupBy** list, select an attribute for sorting issues in all visible folders into groups. The default grouping is Category.

2. Click a tab grouped by issue severity to view the associated issues.
   - The **Critical** tab contains issues that have a high impact and a high likelihood of exploitation. We recommend that you remediate critical issues immediately.
   - The **High** tab contains issues that have a high impact and a low likelihood of exploitation. We recommend that you remediate high issues with the next patch release.
   - The **Medium** tab contains issues that a have low impact and a high likelihood of exploitation. We recommend that you remediate medium issues as time permits.
   - The **Low** tab contains issues that have a low impact and a low likelihood of exploitation. We recommend that you remediate low issues as time permits (your organization can customize this category).
   - The **All** tab contains all issues.

   Within each tab, issues are grouped by issue category. After each grouping name, enclosed in brackets, is the number of audited issues and the total number of issues in the group.

3. To view removed and/or suppressed issues,
   click **Removed** and/or **Suppressed** from the **Visibility** list.

4. Click to expand a grouping and view the issues it contains.

   The Fortify Code Security Extension for Visual Studio Code retrieves the corresponding issues from Application Security Center.

5. Select an issue to view its details in the **ISSUE AUDIT** panel.

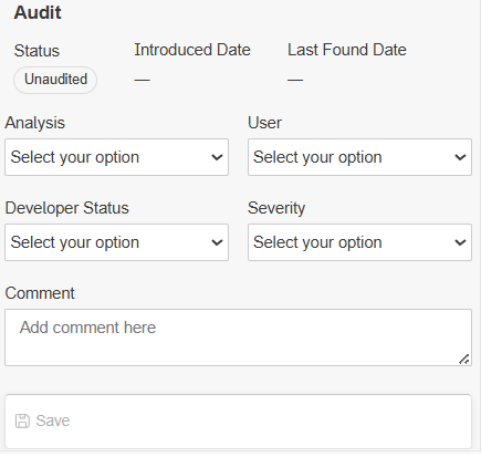
6. Click the **Refresh** icon to sync any changes to the issues in the application version in Application Security.

Viewing issue information

After you select an issue in the **ISSUE AUDIT** panel, the Fortify Code Security Extension for Visual Studio Code displays the issue-specific content in the in the **ISSUE AUDIT** panel on the **Audit**, **Analysis Trace**, **Vulnerablities**, **Recommendations**, and **History** tabs.

Auditing issues

The **AUDIT** view provides a dashboard of analysis information for the selected issue.



To audit an issue:

1. Make sure that the **Audit** view is open.
2. From the issues list in the **REMEDIATE** section, select an issue. You can select multiple issues to make the same edits to multiple issues.
3. In the **Audit** view, select the user to assign to the issue from **User** list.
4. To change the analysis status, select the status from the **Analysis** list.
5. To add a comment for the issue, type your comment in the box at the bottom of the **Comments** area..
6. Click **Save**.

   The changes and comments are displayed in the **History** tab.

The Fortify Code Security Extension for Visual Studio Code saves your changes for the selected application version.
Any changes you make on the **Audit** tab are automatically uploaded to the application version in Application Security Center.

Grouping issues

The following grouping options are available on the **REMEDIATE** section:

Group	Description
Analysis	Issues that have the specified audit analysis value such as `exploitable`, `not an issue`, and so on.
Analysis type	Issues based on the analyzer product.
Analyzer	Issues for the specified analyzer such as `control flow`, `data flow`, `structural`, and so on.
App Defender Protected	App Defender Protected issues.
Audience	Issues by intended audience.
Audited	Issues to find `true` if the primary custom tag is set and `false` if the primary custom tag is not set. The default primary tag is the Analysis tag.
Category	The given category or category substring.
Engine Priority	issues based on the original priority value determined by the engine that identified the issue.
File Name	Issues where the primary location or sink node function call occurs in the specified file.
Folder	Issues based on their assignment to a Fortify folder.

Group	Description
Fortify Priority Order	Issues that have a priority level that matches the specified priority.
Issue State	The issue state, which is `new`, `updated`, `reintroduced`, or `removed`.
Kingdom	All issues in the specified kingdom.
Manual	
<metadata_listname>	Issues with the specified metadata external list. Metadata external lists include `[OWASP top ten <year>]`, `[CWE top 25 <version>]`, `[stig <version>]`, and `[pci ssf <version>]`, and others.
Package	Issues where the primary location occurs in the specified package or namespace. For dataflow issues, the primary location is the sink function.
Primary Context	Issues where the primary location or sink node function call occurs in the specified code context.
Priority Override	

Group	Description
Sink	Issues that have the specified sink function name.
Source	Dataflow issues that have the specified source function name.
Source Context	Dataflow issues that have the source function call contained in the specified code context
Source File	Dataflow issues with the source function call that the specified file contains.
Status	Issues that have the status reviewed, not reviewed, or under review.
Taint Flag	Issues that have the specified taint flag.

Analysis Trace

The **Analysis Trace** tab displays the relevant trace output. This is a set of program points that show how the analyzer found the issue. VS Code places the focus on the line of code that contains the selected security-related issue.

For dataflow issues, this trace is a presentation of the path that the tainted data follows from the source function to the sink function. For example, when you select an issue that is related to potentially tainted dataflow, the analysis trace box shows the direction of the dataflow in this section of the source code.

Analysis Trace Icons

The analysis trace icons described in the following table show how dataflow moves in the section of the source code or execution order.

Icon	Description	Icon	Description
	Data is assigned to a field or variable		Tainted data is returned from a function
	Information is read from a source external to the code such as an HTML form or a URL		A pointer is created
	Data is assigned to a globally scoped field or variable		A pointer is dereferenced
	A comparison is made		The scope of a variable ends
	The function call receives tainted data		The execution jumps
	The function call returns tainted data		A branch is taken in the code execution

Icon	Description	Icon	Description
	Passthrough, tainted data passes from one place to another This is typically shown as functionA(x : y) to indicate that data is transferred from x to y. The x and y values are one of the following: • An argument index • `return` —The return value of a function • `this` —The instance of the current object • A specific object field or key		A branch is not taken in the code execution
	An alias is created for a memory location		Generic

Icon	Description	Icon	Description
	Data is read from a variable		A runtime source, sink, or validation step
	Data is read from a global variable		Taint change

Vulnerability

The **Vulnerability** tab displays the following technical details about the issue: issue overview; issue name; file name and path; instance ID and primary rule GUID; package name; probability; references; and link to Secure Code Warrior.

1.4.4. Auto-remediate your code in Application Security Center

Use the **Auto Remediate** step in an Application Security Center workflow to apply SAST Aviator remediation guidance directly to the source code in your current workspace.

> **Note**
> This workflow is intended for Application Security Center application versions that already contain a SAST Aviator-processed artifact.

When you run the Auto Remediate step, the extension uses `fcli aviator ssc apply-remediations` in the background to retrieve the artifact and apply the available code changes directly to the local source directory.

Requirements

Before you begin, make sure that the following requirements are met:

- An active Application Security Center session exists.
- A workspace folder that contains the application source code is open in Visual Studio Code.
- An Application Security Center application version has already been selected in the sidebar.
- Already performed the SAST Aviator audit on the selected application version.

Perform auto-remediation

To auto-remediate vulnerabilities in an Application Security Center application version:

1. In the extension sidebar, complete the Application Security Center workflow steps to log in and select the target application version.
2. In the sidebar, select **Auto Remediate**.
3. Click **Aviator Authentication** to authenticate your Application Security Center Aviator account and provide the relevant credentials described in the following table.

Field	Description
Aviator URL	(Required) Specifies the SAST Aviator server URL.
User Access Token	(Required) Specifies the SAST Aviator personal access token for the user.

4. Click **Authenticate**.

   Upon successful authentication, **Authenticated** status is displayed. Otherwise, the status displays **Not Authenticated**.

   The **Aviator Authentication** remains valid even if you restart the extension or VS Code and remains active until the session expires.

5. (Optional) Select the **Run Audit before Remediation** check box to trigger a SAST Aviator audit in an application.

   Recommendation

   Select this option when you want remediation recommendations to be based on the most recent scanned version of the code, rather than using a previously audited artifact.
   - When enabled, the extension runs a SAST Aviator audit for the selected Application Security Center application version, waits for the audited artifact to become available, and then applies the remediation suggestions.
   - When disabled, the extension skips the audit step and applies remediations from the most recent eligible SAST Aviator–processed artifact already available in Application Security Center. This option completes faster, but the remediation suggestions may be based on an older scan context.

6. (Required when you select **Run Audit before Remediation**) Specify the target application name in SAST Aviator in the **Aviator Application Name** box.

   Note

   The **Aviator Application Name** identifies the target application in SAST Aviator that is used for the audit and remediation workflow. If the name is incorrect or does not exist in SAST Aviator, the workflow can fail during audit or remediation retrieval, so verify the exact SAST Aviator application name before running.

7. Click **Auto Remediate** to apply the remediations directly to the local source files for the selected application version.

Tip

Review the updated source files and inspect the generated difference in a version control system before committing the changes. After the changes are committed, you can review and test the code, then run another scan to verify that the vulnerabilities were resolved.

Note

If no SAST Aviator remediation data is available or if the selected artifact is not eligible for auto-remediation, the extension shows an error or informational message so that you can correct the prerequisite and try again.
Auto-remediation availability depends on SAST Aviator support for the language, file type, and vulnerability category in the analyzed artifact. You must review the generated changes before committing them to your code base.

1.5. Using Fortify CLI from the Command Palette

The extension uses fcli as a unified command-line interface to work with both Core Application Security (Fortify on Demand) and OpenText Application Security Center (Software Security Center). Instead of handling platform-specific behavior within the extension, fcli provides a consistent command set by abstracting platform differences such as authentication, application structure, and scan submission.

When you set the preferred platform in the extension settings, the extension automatically invokes the corresponding fcli module (fod or ssc). Session handling is managed automatically and results are normalized so that scanning, issue retrieval, and remediation workflows behave consistently, regardless of the selected platform.

Using quick links

The extension provides platform-specific **Quick Links** in the sidebar to help you perform common tasks quickly without typing commands manually.

FoD quick links

The following quick links are available for Core Application Security (Fortify on Demand)

- **Authenticate**: Authenticates to Core Application Security (Fortify on Demand) using fcli. You must authenticate before performing Core Application Security (Fortify on Demand) operations such as viewing issues or submitting scans.
- **Issue list**: Displays vulnerability issues for the selected Core Application Security (Fortify on Demand) application, release, or scan.
- **Remote scan**: Submits a remote scan request to Core Application Security (Fortify on Demand). Use this option to configure and start scans directly from the extension.

SSC quick links

The following quick links are available for OpenText Application Security Center (Software Security Center):

- **Authenticate**: Authenticates to OpenText Application Security Center using fcli to enable access to applications and scan results.

- **Issue list**: Displays vulnerability issues for the selected OpenText Application Security Center application version.

Discover commands dynamically

Use `FCli: Discover Resources and Commands` to view the parameters and options supported by your installed fcli version, rather than entering commands manually.

When you click `FCli: Discover Resources and Commands`:

- The extension queries fcli for all available commands for the preferred platform.
- fcli returns platform-specific commands, parameters, and valid options in JSON-RPC format.
- Use the **Execute via Dialog** option to specify command parameters in a form. The commands to execute are automatically updated as you specify the command parameters in the form.

Example

For Core Application Security (Fortify on Demand), fcli exposes commands such as:

- `fcli fod sast-scan start` : Submit a remote SAST scan
- `fcli fod issue list` : Retrieve vulnerabilities
- `fcli fod release get` : Fetch release details

Example

For Application Security, fcli exposes command such as:

- `fcli ssc artifact upload` : Upload scan results
- `fcli ssc issue list` : Retrieve vulnerabilities
- `fcli ssc appversion get` : Fetch application version details

Save configuration in Favorites

Use the **Favorites** section to save and reuse command configurations for recurring workflows.

1. Open the **Fortify Code Security** activity bar view.
2. Select **View > Command Palette...** .
3. In the search box, type **>FCli: Discover Resources and Commands**.

   Based on the preferred platform, fcli displays all the available commands for the platform and categorizes the commands based on the resources.

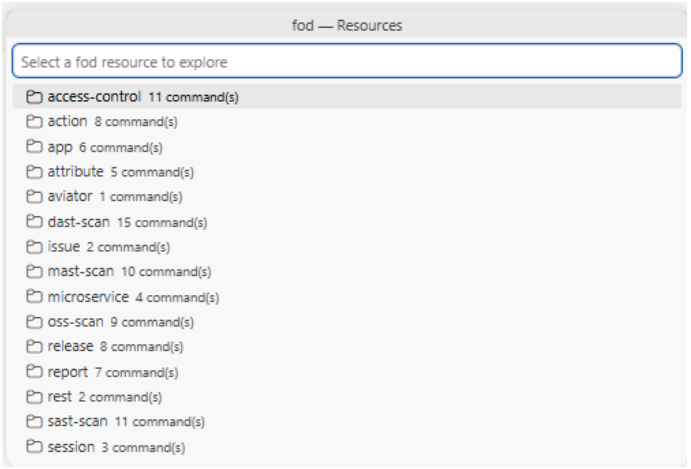
4. Select a command from the list.

5. Select **Execute via Dialog** to open a dialog box for the base fcli command you want to save.
6. Edit the parameters in the dialog box.
7. Select **Save Configuration**.
8. Enter a configuration name in the Save Configuration dialog box.
9. Press **Enter** key to confirm.
10. Expand **Favorites** in the sidebar. The configuration is saved under the folder name corresponding to the base fcli command.
11. Select the saved configuration.
12. Run, update, rename, or delete the saved configuration using inline actions.

Example: Creating a Core Application Security (Fortify on Demand) application

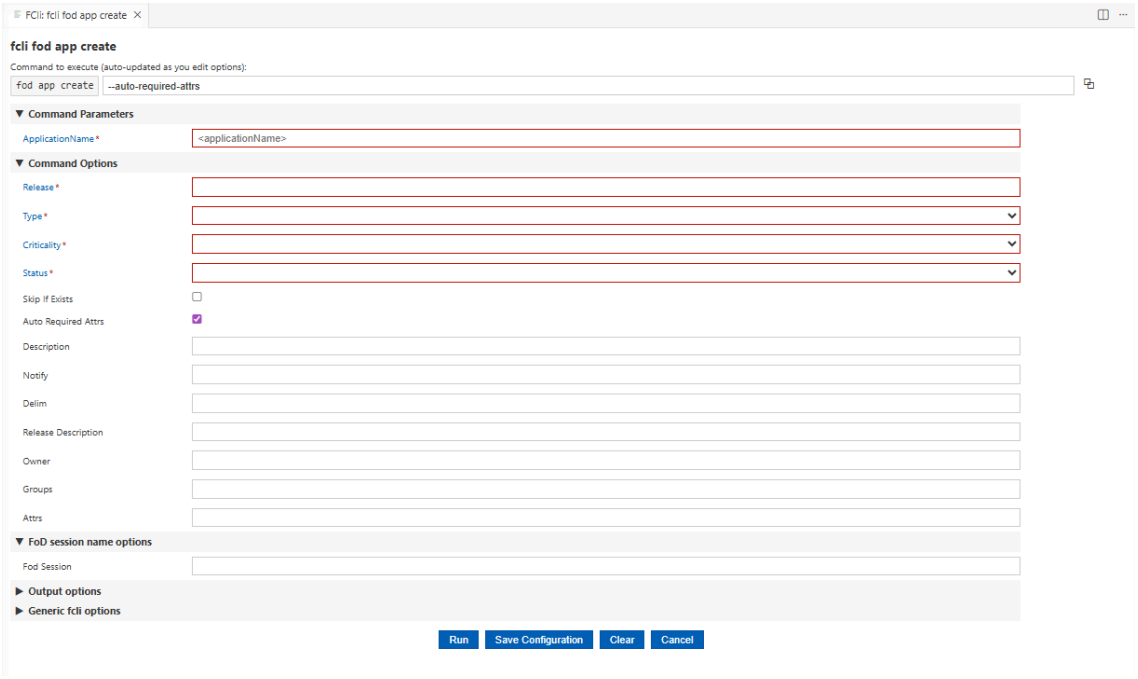
1. Open the **Fortify Code Security** activity bar view.
2. In the **VS Code Settings**, select **Core Application Security (FoD)** as the **Preferred Platform**.
3. Select **View > Command Palette...** .
4. In the search box, type **>FCli: Discover Resources and Commands**.
5. Select **FCli: Discover Resources and Commands > fod**

   fcli displays all the available commands for Core Application Security (Fortify on Demand) and categorizes the commands based on the resources.



6. Select **app > fcli fod app create** to create a new application.
7. Select either **Execute via Dialog**, to open a command dialog with all the parameters and options, or **Execute with Arguments**, to manually type the arguments and execute.

    1. When you select the **Execute via Dialog** option, fcli displays the form for `fcli fod app create` .



1. Specify the Application Name in the **Command Parameters** area.
2. Specify the Release, Type, Criticality, Status, and other advanced options in the **Command Options** area.

   When you edit a parameter or option, fcli automatically updates the command that will be executed in the **Commands to execute** area for `fcli fod app create`. You can copy the full command to run a direct command.
3. Click **Run** to run the command.
4. (Optional) Click **Save Configuration** to save the configuration in **Favourites** in the extension sidebar.

2. When you select the **Execute with Arguments** option, fcli display a text box to run a direct command with explicit arguments when you already know the values.
   1. Specify the arguments for the command in the text box.
   2. Press **Enter** key to confirm and run the command.

When you run the command, fcli sends a request to create the application in your Core Application Security (Fortify on Demand) tenant. The confirmation message is displayed in the OUTPUT panel in JSON-RPC channel indicating that the application release were created successfully.

The new application is immediately available in your Core Application Security (Fortify on Demand) instance and appears in the extension during the **SELECT**

APPLICATION step. You can then select it to start **START SCAN** or **REMEDIATE** workflows. If a validation error occurs, such as a duplicate application name or insufficient permissions, fcli displays an error message in the OUTPUT console so you can correct the issue and try again.

1.6. Using GitHub Copilot with the Fortify Code Security Extension

Use GitHub Copilot Chat with Fortify Code Security Extension for Visual Studio Code to investigate findings, retrieve vulnerability data from Fortify platforms, and accelerate remediation decisions from inside Visual Studio Code.

The extension connects GitHub Copilot to Fortify in two ways:

- **Skills integration**: The extension contributes Fortify skill files that guide GitHub Copilot through common workflows for fcli, Application Security Center, Core Application Security (Fortify on Demand), CI-CD integration, and remediation.
- **MCP server integration**: The extension can register MCP servers through fcli so GitHub Copilot can call platform-aware tools in real time. Click the **Configure Tools** icon in GitHub Copilot Chat and select only the necessary tools to perform the required action. For more information about using MCP servers, see *Visual Studio Code Documentation*.

Tip
Select only the tools that are relevant for your prompt to improve your results.

Requirements

Before using GitHub Copilot for Fortify tasks, you must meet the following requirements:

- GitHub Copilot Chat is enabled in Visual Studio Code.
- The extension is configured with a valid fcli executable path.
- Configure the extension setting **Fortify: Mcp Skill Preference** to control how the `@fortify` experience is routed.
- You are logged in to Application Security Center or Core Application Security (Fortify on Demand).
- If you want MCP-based operations, ensure MCP support is enabled in Visual Studio Code settings.

Note
This setting is disabled when MCP is turned off globally in Visual Studio Code settings or by your organization's policies.

Enable MCP server tools

1. Set the extension setting **Fortify: Mcp Skill Preference** to **MCP**.
2. Select the **Configure Tools** button in the chat input to see all available tools for the MCP server.
3. For Application Security Center workflows related tools, click **fortifySsc > Update Tools**
4. For Core Application Security workflows related tools, click **fortifyFod > Update Tools**

Tip

Enable only the tools you need and leave the others unselected. Selecting unnecessary tools can reduce accuracy and cause the Copilot Agent to hallucinate or perform poorly.

Copilot Chat participants and usage

- **@fortify** : Use `@fortify` for Fortify platform operations and workflow assistance in Application Security Center, and Core Application Security (Fortify on Demand) such as:
  - Resolve application and version or release context from the extension sidebar selection.
  - Detects missing authentication and dynamically provides a sign-in link for the required platform, so you can open the login screen directly from chat and continue the request after authentication.
  - Retrieve vulnerabilities for selected or explicitly named application version or release.
  - Filter vulnerability results by severity, category, audit status, suppression state, and keyword.
  - Provide Fortify-aware remediation guidance and next-step recommendations.
  - Assist with workflow tasks such as scan-related guidance, session checks, and command construction using fcli patterns.

Example

You can use prompts such as:

- @fortify Show High and Critical vulnerabilities for Payments:main in SSC.
- @fortify List SQL Injection issues for RetailBank release main in FoD.
- @fortify Get vulnerabilities for the currently selected application version and include suppressed issues.
- @fortify What should I fix first and why for this application version?

- **@vulnerabilities** : Use `@vulnerabilities` language model tool to analyze vulnerability details and get code-focused remediation help inside Copilot Chat such as:
  - Explain why a finding is risky and how it can be exploited.
  - Summarize likely root causes and affected code paths.
  - Suggest secure fix patterns and validation or sanitization approaches.
  - Help prioritize findings by risk and remediation effort.
  - Generate developer-friendly remediation explanations you can use in review discussions.

Note

If Language Model Tool usage is disabled by your organization, the @vulnerabilties tool cannot be used.

Example

You can use prompts such as:

- @vulnerabilities Explain this finding and propose the safest fix approach.
- @vulnerabilities Compare two remediation options and list trade-offs.
- @vulnerabilities Draft a secure coding checklist for recurring injection issues.

Recommendation

Recommended usage pattern:

- Use @fortify to pull current findings for the target Application Security version or Core Application Security (Fortify on Demand) release.
- Use @vulnerabilities to break down specific issues and shape remediation strategy.
- Apply code fixes in the workspace and validate with your normal build and test process.
- Re-run analysis to confirm the issues are resolved.

© Copyright 2026 Open Text

For more info, visit https://docs.microfocus.com