Lab requirements

- This lab requires **Microsoft Edge** or an Azure DevOps supported browser.
- **Set up an Azure DevOps organization:** If you don't already have an Azure DevOps organization that you can use for this lab, create one by following the instructions available at Create an organization or project collection.

Lab overview

In this lab, you'll use **WhiteSource Bolt with Azure DevOps** to automatically detect vulnerable open source components, outdated libraries, and license compliance issues in your code. You'll use WebGoat, an intentionally insecure web application maintained by OWASP designed to illustrate common web application security issues.

WhiteSource is the leader in continuous open source software security and compliance management. WhiteSource integrates into your build process, irrespective of your programming languages, build tools, or development environments. It works automatically, continuously, and silently in the background, checking your open source components' security, licensing, and quality against WhiteSource constantly updated definitive database of open source repositories.

WhiteSource provides WhiteSource Bolt, a lightweight open source security and management solution developed specifically for integrating Azure DevOps.

Note: WhiteSource Bolt works per project and doesn't offer real-time alert capabilities, which requires a **Full platform**.

WhiteSource Bolt generally is recommended for larger development teams that want to automate their open source management throughout the entire software development lifecycle (from the repositories to post-deployment stages) and across all projects and products.

Azure DevOps integration with WhiteSource Bolt will enable you to:

- Detect and remedy vulnerable open source components.
- Generate comprehensive open source inventory reports per project or build.
- Enforce open source license compliance, including dependencies' licenses.
- Identify outdated open source libraries with recommendations to update.

Objectives

After you complete this lab, you will be able to:

- Activate WhiteSource Bolt.
- Run a build pipeline and review the WhiteSource security and compliance report.

Estimated timing: 45 minutes

Instructions

Exercise 0: Configure the lab prerequisites

In this exercise, you will set up the prerequisites for the lab, which consist of a new Azure DevOps project with a repository based on the Parts Unlimited MRP GitHub repository.

Task 1: Create and configure the team project

In this task, you will use Azure DevOps Demo Generator to generate a new project based on the WhiteSource-Bolt template

1. On your lab computer, start a web browser and navigate to Azure DevOps Demo Generator. This utility site will automate the process of creating a new Azure DevOps project within your account that is prepopulated with content (work items, repos, etc.) required for the lab.

2. **Note**: For more information on the site, see https://docs.microsoft.com/en-us/azure/devops/demo-gen.

3. Click **Sign in** and sign in using the Microsoft account associated with your Azure DevOps subscription.
4. If required, on the **Azure DevOps Demo Generator** page, click **Accept** to accept the permission requests for accessing your Azure DevOps subscription.
5. On the **Create New Project** page, in the **New Project Name** textbox, type **WhiteSource Bolt**, in the **Select organization** dropdown list, select your Azure DevOps organization, and then click **Choose template**.
6. In the list of templates, in the toolbar, click **DevOps Labs**, select the **WhiteSource Bolt** template and click **Select Template**.
7. Back on the **Create New Project** page, if prompted to install a missing extension, select the checkbox below the **WhiteSource Bolt** and click **Create Project**.

8. **Note**: Wait for the process to complete. This should take about 2 minutes. In case the process fails, navigate to your DevOps organization, delete the project, and try again.

9. On the **Create New Project** page, click **Navigate to project**.

Exercise 1: Implement Security and Compliance in an Azure Pipeline using WhiteSource Bolt

In this exercise, leverage WhiteSource Bolt to scan the project code for security vulnerabilities and licensing compliance issues, and view the resulting report.

Task 1: Activate WhiteSource Bolt

In this task, you will activate WhiteSource Bolt in the newly generated Azure Devops project.

10. On your lab computer, in the web browser window displaying the Azure DevOps portal with the **WhiteSource Bolt** project open, **in the vertical menu bar** at the far left of the

Azure DevOps portal, click **Pipelines** section and **WhiteSource Bolt** option (in the vertical menu bar under "Deployment Groups" option).

11. On the **You're almost there** pane, provide your **Work Email** and **Company Name**, in the **Country** dropdown list, select the entry representing your country, and click *Get Started* button to start using the *Free* version of WhiteSource Bolt. This will automatically open a new browser tab displaying the **Get Started With Bolt** page.
12. Switch back to the web browser tab displaying the Azure DevOps portal and verify that the **You are using a FREE version of WhiteSource Bolt** is displayed.

Task 2: Trigger a build

In this task, you will trigger a build within your Java code-based Azure DevOps project. You will use **WhiteSource Bolt** extension to identify vulnerable components present in this code.

13. On your lab computer, in the vertical menu bar on the left side, navigate to the **Pipelines** section, click **WhileSourceBolt**, click **Run pipeline** and then, on the **Run pipeline** pane, click **Run**.
14. On the **Summary** tab of the build pane, in the **Jobs** section, click **Phase 1** and monitor the progress of the build process.

15. **Note**: The build may take a few minutes to complete. The build definition consists of the following tasks:

Tasks	Usage
npm	Installs and publishes npm packages required for the build
Maven	builds Java code with the provided pom xml file
WhiteSource Bolt	scans the code in the provided working directory/root directory to detect security vulnerabilities, problematic open source licenses
Copy Files	copies the resulting JAR files from the source to the destination folder using match patterns
Publish Build Artifacts	publishes the artifacts produced by the build

16. Once the build completes, navigate back to the **Summary** tab and review **Tests and coverage** section.

Task 3: Analyze Reports

In this task, you will review the WhiteSource Bolt build report.

17. On the build pane, click the **WhiteSource Bolt Build Report** tab header and wait for the report to fully render.
18. While on the **WhiteSource Bolt Build Report** tab, verify that WhiteSource Bolt automatically detected Open Source components in the software including transitive dependencies and their respective licenses.

19. While on the **WhiteSource Bolt Build Report** tab, review the Security dashboard, displaying the vulnerabilities discovered during the build.

20. **Note**: The report displays the list of all vulnerable open source components, including **Vulnerability Score**, **Vulnerable Libraries**, and **Severity Distribution**. You can identify the opensource license distribution by leveraging a detailed view of all components and links to their metadata and licensed references.

21. While on the **WhiteSource Bolt Build Report** tab, scroll down to the **Outdated Libraries** section and review its content.

22. **Note**: WhiteSource Bolt tracks outdated libraries in the project, providing library details, links to newer versions, and remediation recommendations.

Review

In this lab, you will use **WhiteSource Bolt with Azure DevOps** to automatically detect vulnerable open source components, outdated libraries, and license compliance issues in your code.