YouTestMe

Third-Party Dependency Management

Table of Contents

1 Security, Vulnerability Library Scanning Procedure

1.1 Introduction

The security and reliability of YouTestMe releases are ensured by regularly scanning project dependencies for vulnerabilities and identifying any end-of-life (EOL) or end-of-service (EOS) issues.

This procedure is referenced in the Release Procedure.

1.2 Tools Used

1.2.1 OWASP Dependency-Check

- Identifies vulnerabilities in dependencies.
- Generates detailed HTML reports.
- Integrated via the release pom.xml file (i.e. https://svn.youtestme.com/scm/svn/getcertified/branches/GC-7.3/www_source/ytm.model/pom.xml)

1.2.2 Snyk

- Provides comprehensive console output and reports.
- Ensures dependencies are secure and up-to-date.
- Seamlessly integrates with Maven projects.

1.2.3 Maven plugin

- Generates dependency tree License report, suitable for manual checking of each library and its license.

1.3 OWASP

1.3.1 OWASP Dependency-Check Integration

1. **Add OWASP Dependency-Check Plugin to pom.xml**

Add the following plugin configuration to the parent project's pom.xml file to integrate OWASP Dependency-Check:

```
xml
Copy code
<build>
    <plugins>
        <plugin>
            <groupId>org.owasp</groupId>
            <artifactId>dependency-check-maven</artifactId>
            <version>6.5.0</version>
            <executions>
                <execution>
                    <goals>
                        <goal>check</goal>
                    </goals>
                </execution>
            </executions>
        </plugin>
    </plugins>
</build>
```

2. **Configure Output (Optional)**

Configure the output format and directory as follows:

```
xml
Copy code
<configuration>
    <format>HTML</format>
    <outputDirectory>${project.build.directory}/dependency-check-
report</outputDirectory>
</configuration>
```

3. **Run the Scan**

Execute the OWASP Dependency-Check scan by running:

```
bash
Copy code
mvn verify
```

The report is generated in the specified output directory.

1.4 Snyk

1.4.1 Installation

1. **Install and Authenticate Snyk CLI**

Install the Snyk CLI globally using npm:

```
bash
Copy code
npm install -g snyk
```

Authenticate the Snyk CLI:

```
bash
Copy code
snyk auth
```

2. **Add Snyk Maven Plugin to pom.xml**

Include the Snyk Maven plugin configuration in the parent project's pom.xml file:

```
xml
Copy code
<build>
    <plugins>
        <plugin>
            <groupId>io.snyk</groupId>
            <artifactId>snyk-maven-plugin</artifactId>
            <version>2.0.0</version>
            <executions>
                <execution>
                    <goals>
                        <goal>test</goal>
                    </goals>
                </execution>
            </executions>
        </plugin>
    </plugins>
</build>
```

3. **Run the Scan**

Execute the Snyk scan by running:

```
bash
Copy code
mvn snyk:test
```

The results are displayed in the console output.

1.4.2 Snyk Sample from YouTestMe GetCertified

Snyk produces a file with directions for fixing libraries:

```
[INFO] Tested 78 dependencies for known issues, found 14 issues, 14 vulnerable
paths.
[INFO]
[INFO]
[INFO] Issues to fix by upgrading:
[INFO]
[INFO]   Upgrade com.google.cloud:google-cloud-translate@2.1.12 to
com.google.cloud:google-cloud-translate@2.20.0 to fix
[INFO]   â?? Creation of Temporary File in Directory with Insecure Permissions
[Low Severity][https://security.snyk.io/vuln/SNYK-JAVA-COMGOOGLEGUAVA-5710356]
in com.google.guava:guava@31.1-jre
[INFO]     introduced by com.google.cloud:google-cloud-translate@2.1.12 >
com.google.guava:guava@31.1-jre
[INFO]   â?? Denial of Service (DoS) [Medium
Severity][https://security.snyk.io/vuln/SNYK-JAVA-COMGOOGLEPROTOBUF-3040284] in
com.google.protobuf:protobuf-java@3.19.4
```

1.4.3 Post Scan

Create vulnerability report for management.

1.5 Maven dependency scan

1.5.1 Generate Dependency Tree in Text Format

1. Open your terminal or command prompt.
2. Navigate to the root directory of your Maven project.
3. Run the following command to generate the dependency tree in text format:

```
mvn dependency:tree -DoutputType=text -DoutputFile=dependency-tree.txt
```

1.5.2 Convert Text to HTML

If you haven't already, install Pandoc. You can download it from Pandoc's website and follow the installation instructions for your operating system.

Once Pandoc is installed, navigate to the directory where you saved the dependency-tree.txt file.

Run the following command to convert the text file to HTML format:

```
pandoc -f plain -t html dependency-tree.txt -o dependency-tree.html
```

1.5.3 Include in Maven Site

Open your project's pom.xml file in a text editor.

Inside the <build> section, add the following plugin configuration:

```
<plugins>
    <plugin>
        <groupId>org.apache.maven.plugins</groupId>
        <artifactId>maven-site-plugin</artifactId>
        <version>3.14</version>
        <configuration>
            <reports>
                <report>
                    <groupId>org.apache.maven.plugins</groupId>
                    <artifactId>maven-project-info-reports-plugin</artifactId>
                    <version>3.1.2</version>
                    <reportSets>
                        <reportSet>
                            <reports>
                                <report>dependency-convergence</report>
                                <report>dependencies</report>
                            </reports>
                        </reportSet>
                    </reportSets>
                </report>
            </reports>
        </configuration>
    </plugin>
</plugins>
```

1.5.4 Generate Maven Site

1. Save the changes to your pom.xml file.
2. In the terminal or command prompt, run the following command to generate the Maven site:

```
mvn site
```

3. After the command completes, navigate to the target/site directory in your project. You should find the Maven site generated, including the HTML dependency report (dependency-tree.html).

1.5.5 Manual Checks for EOL/EOS:

Some libraries may fade without public announcements. Check relevant websites.

Manually check library websites for EOL/EOS status.

1.6 Post-Scan Actions

1.6.1 Report Examination and Prioritization

1. Thoroughly review generated reports.
2. Categorize libraries by priority for upgrades:
    a. **Critical Issues**: Immediate attention.
    b. **High Severity**: Addressed in the next sprint.
    c. **Medium Severity**: Planned for subsequent sprints.
    d. **Low Severity**: Monitored based on impact.
3. Consider public and software recommendations for remediation.
4. Produce a report for CTO, Product Management, and Release Management with reasons and recommendations for using EOL/EOS libraries.
5. Place the report in the relevant release documentation folder (i.e., https://svn.youtestme.com/scm/svn/youtestmedoc/trunk/Projects/YTM Get Certified/Releases/Archive/Release 7.0.0r May-5-2019)

1.7 Issue Analysis and Remediation

1.7.1 Root Cause Understanding

1. Analyze each issue to understand its origin.
2. Make informed decisions based on findings.

1.7.2 Detailed Remediation Plan

3. Create a plan with specific steps:
    a. Update or replace dependencies.
    b. Implement mitigations.
    c. Ensure ongoing system security.

1.7.3 Sign-Off

4. The plan must be signed off for the next release by the CTO and Release Manager.