Hands-on Lab: Using Dynamic Analysis

Estimated time needed: 30 minutes

Welcome to the **Using Dynamic Analysis** lab! In this lab, you will learn how to install, configure, and use OWASP ZAP for dynamic analysis of your project code.

If you ask a developer what their top three goals are, you are most likely going to hear the following:

- Write bug-free code
- Meet design specifications
- Prevent security issues

To meet these goals, development teams often need to review and test their code comprehensively. Code analysis is one of the solutions. In this lab, you will see how using dynamic code analysis can help you prevent security issues.

Learning Objectives

- Install and configure OWASP ZAP
- Use a dynamic analysis tool
- Interpret security reports from ZAP

Why Dynamic Code Analysis?

What is Dynamic Code Analysis?

It is the testing and evaluation of an application during runtime. Also referred to as dynamic code scanning, dynamic analysis can identify security issues that are too complicated for static analysis alone to reveal.

Dynamic application security testing (DAST) looks at the application from the outside-in — by simulating attacks against a web application and analyzing the application's responses to discover security vulnerabilities in the application.

Using OWASP ZAP to Check App Vulnerabilities

In this lab, you will get hands-on experience using OWASP ZAP to conduct dynamic analysis. A real-world application that you will be testing is the OWASP Juice Shop app, which is an application developed for security training purposes. For a detailed introduction, full list of features, and architecture overview, please visit the official project page: https://owasp-juice.shop.

You will be guided through setting up OWASP ZAP and using it to run an analysis of the Juice Shop application in the Cloud IDE with Docker so that everything can be done in a terminal on the right panel. You should be able to replicate this lab easily in any environment with Docker installed, including your developer workstation.

To get a ZAP server up and running in the next steps, you will:

1. Fetch the application that you will scan
2. Run ZAP against the application
3. Interpret the results of the scanning

Step 1: Fetch the Insecure App: Juice Shop

To test the OWASP Juice Shop app, you must fetch and run the application. Open up a terminal by clicking `Terminal -> New Terminal` in the top menu bar. Copy and paste the following commands in the terminal window to fetch Juice Shop's docker image, and then run the application in the current Cloud IDE.

```
docker pull bkimminich/juice-shop
docker run --rm -p 3000:3000 bkimminich/juice-shop
```

After running the two commands, wait until you see the message “info: Server listening on port 3000” in the terminal before proceeding with the lab. If you do not see this message, try leaving this lab and then restarting it.

```
info: Server listening on port 3000
```

In the next step, you will look at the application's web interface.

Step 2: Launch the Juice Shop UI

Next, click the **Web Application** button below. Once you've clicked the button, you will see the app start running!

Web Application

The user interface should look like the following image:

Step 3: Run OWASP ZAP

You are now ready to download and run the ZAP tool in a Docker container.

Your Task

1. Open a new terminal window using `Terminal > New Terminal` to issue new `docker` commands.
2. In the terminal, execute the `docker pull` command to download/pull the docker image of OWASP ZAP. (*Note: It may take some time to download.*)

   ```
   docker pull softwaresecurityproject/zap-stable
   ```

   Now that the tool is installed in the current Cloud IDE, you can start a vulnerability scan of the Juice Shop app.

3. Next, copy the URL of the app from the address bar of the running application in Cloud IDE to your clipboard. Then run the following command and replace the `{TARGET_URL}` with the URL you copied.

   ```
   docker run -t softwaresecurityproject/zap-stable zap-baseline.py -t {TARGET_URL}
   ```

Explanation of the Command

This command initiates a baseline scan of the specified target URL using the OWASP ZAP tool in a Docker container. Here's a breakdown of the command:

- `docker run -t softwaresecurityproject/zap-stable`: This part of the command runs a new Docker container using the `zap-stable` image from the Software Security Project.
- The `-t` option allocates a pseudo-TTY, which is useful for interactive processes.
- `zap-baseline.py`: This is a Python script provided by OWASP ZAP that performs a baseline scan against a web application. The baseline scan is designed to provide a quick assessment of the security of the application by checking for common vulnerabilities.
- `-t <TARGET_URL>`: This option specifies the target URL that you wish to scan. You should replace `<TARGET_URL>` with the actual URL of the web application you are testing (e.g., `http://localhost:3000` for the Juice Shop app).

ZAP will now start its crawling activity of the site and builds a sitemap, and the related output can be reviewed in the terminal. This will take a few minutes to execute.

Results

The output is lengthy, and it will look something like this:

```
Using the Automation Framework
Total of 13 URLs
PASS: Vulnerable JS Library (Powered by Retire.js) [10003]
PASS: In Page Banner Information Leak [10009]
PASS: Cookie No HttpOnly Flag [10010]
```

```
PASS: Cookie Without Secure Flag [10011]
PASS: Content-Type Header Missing [10019]
PASS: Information Disclosure - Debug Error Messages [10023]
PASS: Information Disclosure - Sensitive Information in URL [10024]
PASS: Information Disclosure - Sensitive Information in HTTP Referrer Header [10025]
PASS: HTTP Parameter Override [10026]
PASS: Open Redirect [10028]
PASS: Cookie Poisoning [10029]
PASS: User Controllable Charset [10030]
PASS: User Controllable HTML Element Attribute (Potential XSS) [10031]
PASS: Viewstate [10032]
PASS: Directory Browsing [10033]
PASS: Heartbleed OpenSSL Vulnerability (Indicative) [10034]
PASS: HTTP Server Response Header [10036]
PASS: X-Backend-Server Header Information Leak [10039]
PASS: Secure Pages Include Mixed Content [10040]
PASS: HTTP to HTTPS Insecure Transition in Form Post [10041]
PASS: HTTPS to HTTP Insecure Transition in Form Post [10042]
PASS: User Controllable JavaScript Event (XSS) [10043]
PASS: Big Redirect Detected (Potential Sensitive Information Leak) [10044]
PASS: Retrieved from Cache [10050]
PASS: X-ChromeLogger-Data (XCOLD) Header Information Leak [10052]
PASS: Cookie without SameSite Attribute [10054]
PASS: CSP [10055]
PASS: X-Debug-Token Information Leak [10056]
PASS: Username Hash Found [10057]
PASS: X-AspNet-Version Response Header [10061]
PASS: PII Disclosure [10062]
PASS: Hash Disclosure [10097]
PASS: Source Code Disclosure [10099]
PASS: Weak Authentication Method [10105]
PASS: Reverse Tabnabbing [10108]
PASS: Authentication Request Identified [10111]
PASS: Verification Request Identified [10113]
PASS: Absence of Anti-CSRF Tokens [10202]
PASS: Private IP Disclosure [2]
PASS: Session ID in URL Rewrite [3]
PASS: Stats Passive Scan Rule [50003]
PASS: Insecure JSF ViewState [90001]
PASS: Java Serialization Object [90002]
PASS: Sub Resource Integrity Attribute Missing [90003]
PASS: Insufficient Site Isolation Against Spectre Vulnerability [90004]
PASS: Charset Mismatch [90011]
PASS: Application Error Disclosure [90022]
PASS: WSDL File Detection [90030]
WARN-NEW: Re-examine Cache-control Directives [10015] x 3
        https://manvig1-3000.theiadockernext-1-labs-prod-theiak8s-4-tor01.proxy.cognitiveclass.ai/ (200 OK)
        https://manvig1-3000.theiadockernext-1-labs-prod-theiak8s-4-tor01.proxy.cognitiveclass.ai/robots.txt (200 OK)
        https://manvig1-3000.theiadockernext-1-labs-prod-theiak8s-4-tor01.proxy.cognitiveclass.ai/sitemap.xml (200 OK)
WARN-NEW: Cross-Domain JavaScript Source File Inclusion [10017] x 4
        https://manvig1-3000.theiadockernext-1-labs-prod-theiak8s-4-tor01.proxy.cognitiveclass.ai/ (200 OK)
        https://manvig1-3000.theiadockernext-1-labs-prod-theiak8s-4-tor01.proxy.cognitiveclass.ai/ (200 OK)
        https://manvig1-3000.theiadockernext-1-labs-prod-theiak8s-4-tor01.proxy.cognitiveclass.ai/sitemap.xml (200 OK)
        https://manvig1-3000.theiadockernext-1-labs-prod-theiak8s-4-tor01.proxy.cognitiveclass.ai/sitemap.xml (200 OK)
WARN-NEW: Missing Anti-clickjacking Header [10020] x 2
        https://manvig1-3000.theiadockernext-1-labs-prod-theiak8s-4-tor01.proxy.cognitiveclass.ai/ (200 OK)
        https://manvig1-3000.theiadockernext-1-labs-prod-theiak8s-4-tor01.proxy.cognitiveclass.ai/sitemap.xml (200 OK)
WARN-NEW: X-Content-Type-Options Header Missing [10021] x 9
        https://manvig1-3000.theiadockernext-1-labs-prod-theiak8s-4-tor01.proxy.cognitiveclass.ai/ (200 OK)
        https://manvig1-3000.theiadockernext-1-labs-prod-theiak8s-4-tor01.proxy.cognitiveclass.ai/assets/public/favicon_js.ico (200 OK)
        https://manvig1-3000.theiadockernext-1-labs-prod-theiak8s-4-tor01.proxy.cognitiveclass.ai/main.js (200 OK)
        https://manvig1-3000.theiadockernext-1-labs-prod-theiak8s-4-tor01.proxy.cognitiveclass.ai/polyfills.js (200 OK)
        https://manvig1-3000.theiadockernext-1-labs-prod-theiak8s-4-tor01.proxy.cognitiveclass.ai/robots.txt (200 OK)
WARN-NEW: Information Disclosure - Suspicious Comments [10027] x 2
        https://manvig1-3000.theiadockernext-1-labs-prod-theiak8s-4-tor01.proxy.cognitiveclass.ai/main.js (200 OK)
        https://manvig1-3000.theiadockernext-1-labs-prod-theiak8s-4-tor01.proxy.cognitiveclass.ai/vendor.js (200 OK)
WARN-NEW: Strict-Transport-Security Header Not Set [10035] x 1
        https://manvig1-3000.theiadockernext-1-labs-prod-theiak8s-4-tor01.proxy.cognitiveclass.ai/ftp (400 Bad Request)
WARN-NEW: Server Leaks Information via "X-Powered-By" HTTP Response Header Field(s) [10037] x 9
        https://manvig1-3000.theiadockernext-1-labs-prod-theiak8s-4-tor01.proxy.cognitiveclass.ai/ (200 OK)
        https://manvig1-3000.theiadockernext-1-labs-prod-theiak8s-4-tor01.proxy.cognitiveclass.ai/assets/public/favicon_js.ico (200 OK)
        https://manvig1-3000.theiadockernext-1-labs-prod-theiak8s-4-tor01.proxy.cognitiveclass.ai/main.js (200 OK)
        https://manvig1-3000.theiadockernext-1-labs-prod-theiak8s-4-tor01.proxy.cognitiveclass.ai/polyfills.js (200 OK)
        https://manvig1-3000.theiadockernext-1-labs-prod-theiak8s-4-tor01.proxy.cognitiveclass.ai/robots.txt (200 OK)
WARN-NEW: Content Security Policy (CSP) Header Not Set [10038] x 2
        https://manvig1-3000.theiadockernext-1-labs-prod-theiak8s-4-tor01.proxy.cognitiveclass.ai/ (200 OK)
        https://manvig1-3000.theiadockernext-1-labs-prod-theiak8s-4-tor01.proxy.cognitiveclass.ai/sitemap.xml (200 OK)
WARN-NEW: Non-Storable Content [10049] x 10
        https://manvig1-3000.theiadockernext-1-labs-prod-theiak8s-4-tor01.proxy.cognitiveclass.ai/ftp (400 Bad Request)
        https://manvig1-3000.theiadockernext-1-labs-prod-theiak8s-4-tor01.proxy.cognitiveclass.ai/robots.txt (200 OK)
        https://manvig1-3000.theiadockernext-1-labs-prod-theiak8s-4-tor01.proxy.cognitiveclass.ai/ (200 OK)
        https://manvig1-3000.theiadockernext-1-labs-prod-theiak8s-4-tor01.proxy.cognitiveclass.ai/assets/public/favicon_js.ico (200 OK)
        https://manvig1-3000.theiadockernext-1-labs-prod-theiak8s-4-tor01.proxy.cognitiveclass.ai/main.js (200 OK)
WARN-NEW: Deprecated Feature Policy Header Set [10063] x 6
        https://manvig1-3000.theiadockernext-1-labs-prod-theiak8s-4-tor01.proxy.cognitiveclass.ai/ (200 OK)
        https://manvig1-3000.theiadockernext-1-labs-prod-theiak8s-4-tor01.proxy.cognitiveclass.ai/main.js (200 OK)
        https://manvig1-3000.theiadockernext-1-labs-prod-theiak8s-4-tor01.proxy.cognitiveclass.ai/polyfills.js (200 OK)
        https://manvig1-3000.theiadockernext-1-labs-prod-theiak8s-4-tor01.proxy.cognitiveclass.ai/runtime.js (200 OK)
        https://manvig1-3000.theiadockernext-1-labs-prod-theiak8s-4-tor01.proxy.cognitiveclass.ai/sitemap.xml (200 OK)
WARN-NEW: Timestamp Disclosure - Unix [10096] x 1
        https://manvig1-3000.theiadockernext-1-labs-prod-theiak8s-4-tor01.proxy.cognitiveclass.ai/main.js (200 OK)
WARN-NEW: Cross-Domain Misconfiguration [10098] x 9
        https://manvig1-3000.theiadockernext-1-labs-prod-theiak8s-4-tor01.proxy.cognitiveclass.ai/ (200 OK)
        https://manvig1-3000.theiadockernext-1-labs-prod-theiak8s-4-tor01.proxy.cognitiveclass.ai/assets/public/favicon_js.ico (200 OK)
        https://manvig1-3000.theiadockernext-1-labs-prod-theiak8s-4-tor01.proxy.cognitiveclass.ai/main.js (200 OK)
        https://manvig1-3000.theiadockernext-1-labs-prod-theiak8s-4-tor01.proxy.cognitiveclass.ai/polyfills.js (200 OK)
        https://manvig1-3000.theiadockernext-1-labs-prod-theiak8s-4-tor01.proxy.cognitiveclass.ai/robots.txt (200 OK)
WARN-NEW: Modern Web Application [10109] x 2
        https://manvig1-3000.theiadockernext-1-labs-prod-theiak8s-4-tor01.proxy.cognitiveclass.ai/ (200 OK)
        https://manvig1-3000.theiadockernext-1-labs-prod-theiak8s-4-tor01.proxy.cognitiveclass.ai/sitemap.xml (200 OK)
WARN-NEW: Dangerous JS Functions [10110] x 2
        https://manvig1-3000.theiadockernext-1-labs-prod-theiak8s-4-tor01.proxy.cognitiveclass.ai/main.js (200 OK)
        https://manvig1-3000.theiadockernext-1-labs-prod-theiak8s-4-tor01.proxy.cognitiveclass.ai/vendor.js (200 OK)
WARN-NEW: Session Management Response Identified [10112] x 6
        https://manvig1-3000.theiadockernext-1-labs-prod-theiak8s-4-tor01.proxy.cognitiveclass.ai/ (200 OK)
        https://manvig1-3000.theiadockernext-1-labs-prod-theiak8s-4-tor01.proxy.cognitiveclass.ai/ (200 OK)
        https://manvig1-3000.theiadockernext-1-labs-prod-theiak8s-4-tor01.proxy.cognitiveclass.ai/robots.txt (200 OK)
        https://manvig1-3000.theiadockernext-1-labs-prod-theiak8s-4-tor01.proxy.cognitiveclass.ai/sitemap.xml (200 OK)
        https://manvig1-3000.theiadockernext-1-labs-prod-theiak8s-4-tor01.proxy.cognitiveclass.ai/main.js (200 OK)
WARN-NEW: Loosely Scoped Cookie [90033] x 4
        https://manvig1-3000.theiadockernext-1-labs-prod-theiak8s-4-tor01.proxy.cognitiveclass.ai/ (200 OK)
        https://manvig1-3000.theiadockernext-1-labs-prod-theiak8s-4-tor01.proxy.cognitiveclass.ai/ (200 OK)
        https://manvig1-3000.theiadockernext-1-labs-prod-theiak8s-4-tor01.proxy.cognitiveclass.ai/robots.txt (200 OK)
        https://manvig1-3000.theiadockernext-1-labs-prod-theiak8s-4-tor01.proxy.cognitiveclass.ai/sitemap.xml (200 OK)
FAIL-NEW: 0     FAIL-INPROG: 0  WARN-NEW: 16    WARN-INPROG: 0  INFO: 0 IGNORE: 0       PASS: 48
```

Step 4: Interpret the scan results

Let's look at the reults that came back from the scan.

A number of items tested came back with `PASS:` so we will not look at those.

Our attention is on any warnings or failures. Below is a summary of the ten warnings that came back. Each describes the vulnerability, then cites the number of times it was found (for example, `x 3`), and then lists the URLs that had the vulnerability. We have removed the URLs to create the summary list below:

```
WARN-NEW: Re-examine Cache-control Directives [10015] x 3
WARN-NEW: Cross-Domain JavaScript Source File Inclusion [10017] x 4
WARN-NEW: Missing Anti-clickjacking Header [10020] x 2
WARN-NEW: X-Content-Type-Options Header Missing [10021] x 9
WARN-NEW: Information Disclosure - Suspicious Comments [10027] x 2
WARN-NEW: Strict-Transport-Security Header Not Set [10035] x 1
WARN-NEW: Server Leaks Information via "X-Powered-By" HTTP Response Header Field(s) [10037] x 9
WARN-NEW: Content Security Policy (CSP) Header Not Set [10038] x 2
WARN-NEW: Non-Storable Content [10049] x 10
WARN-NEW: Deprecated Feature Policy Header Set [10063] x 6
WARN-NEW: Timestamp Disclosure - Unix [10096] x 1
WARN-NEW: Cross-Domain Misconfiguration [10098] x 9
WARN-NEW: Modern Web Application [10109] x 2
WARN-NEW: Dangerous JS Functions [10110] x 2
WARN-NEW: Session Management Response Identified [10112] x 6
WARN-NEW: Loosely Scoped Cookie [90033] x 4
FAIL-NEW: 0     FAIL-INPROG: 0  WARN-NEW: 16    WARN-INPROG: 0  INFO: 0 IGNORE: 0       PASS: 48
```

As you can see, this application has vulnerabilities in Cross-Domain JavaScript Source File Inclusion, Missing Anti-clickjacking Header, X-Content-Type-Options Header Missing, Content Security Policy (CSP) Header Not Set, Cross-Domain Misconfiguration, and Loosely Scoped Cookies, just to name a few.

You can use the numbers next to the vulnerability names to read about the alert on the ZAP Proxy Web site. Using the following URL:

```
https://www.zaproxy.org/docs/alerts/{NUMBER}
```

Be sure to replace `{NUMBER}` with the number of the alert.

For example, Cross-Domain JavaScript Source File Inclusion is numbered `10017` in the warning message above, so the URL would be (*click on it and see)*:

https://www.zaproxy.org/docs/alerts/10017

As a developer, your task would be to look up the vulnerability, look at each URL listed as being vulnerable, and then fix the vulnerabilities in the code one by one.

If you run a Dynamic Security Testing (DAST) tool early in your development lifecycle, your list probably won't get as big as this example. Had DAST been done earlier in development, there may not have been nine violations of X-Content-Type-Options Header Missing or Cross-Domain Misconfiguration.

Conclusion

Congratulations! You have completed this lab on dynamic analysis, which is an integral step in secure app development. You are now well on your way to making your applications safer by running dynamic analysis security scans on them.

You now understand the benefits of dynamic analysis and when to use it to detect vulnerabilities in a project. You also know how to get started with the most popular open-source dynamic analysis tool, OWASP ZAP.

Next Steps

Detecting the different vulnerabilities is just one of the first steps in secure app development. You must also understand the meaning behind those vulnerabilities to take corrective actions. There is no better way to learn than by doing.

Your next challenge is to use OWASP ZAP in your development environment to perform security scans on your code and then fix the problems it finds. You are well on your way to writing more secure code.

Author(s)

Roxanne Li
John J. Rofrano

Other Contributor(s)