Article
Henrique Dias · Dec 4, 2021 6m read

Open Exchange

Why? How? What's zap-api-scan-sample?

Hey community, how are you all doing?

What if you could check if your REST application is susceptible to some vulnerability? What if you could check if any known attacks affect your application?

With these issues in mind, we've brought our sample application using the ZAP testing tool. A way to quickly, conveniently provide tools for developers to validate security issues in an accessible manner practically.

Why is it important to validate your application's security?

Increasingly, security weaknesses are being exploited by more and more malicious people. And as developers, providing security for the people who use our application is part of our job.

How to validate?

To make this validation, it is necessary to follow the steps below so that your REST application can take advantage of the features and security tests that we offer with ZAP. See more below.

Why do we choose OWASP?

We chose to use OWASP® Foundation, tests created by experts who maintain a constantly updated vulnerability test repository.

Who is OWASP® Foundation?

The Open Web Application Security Project® (OWASP) is a nonprofit foundation that works to improve the security of software. Through community-led open-source software projects, hundreds of local chapters worldwide, tens of thousands of members, and leading educational and training conferences, the OWASP Foundation is the source for developers and technologists to secure the web.

For our project, we made it using OWASP® Zed Attack Proxy (ZAP), The world's most widely used web app scanner. Free and open source. Actively maintained by a dedicated international team of volunteers.

But let's face it, they lost the chance to name the project after someone who really understands attacks and invasions: Zod, General Zod!!!

How to use ZAP?

Prerequisites

Make sure you have git and Docker desktop installed.

Installation for development with Docker

Clone/git pull the repo into any local directory:

```
$ git clone https://github.com/jrpereirajr/zap-api-scan-sample.git
$ cd zap-api-scan-sample
```

Open the terminal in this directory and run:

```
$ docker-compose up -d --build
```

Note: as in this version a file transfer is used in order to let containers to communicate to each other, it ' s necessary to grant some privileges for writing in the shared volume:

```
$ chmod 777 -R zap-pool
```

Scanning your APIs

This sample lets you scan each REST API or all of them at once.

For instance, if you would like to scan the API */crud*, run this command:

```
Do ##class(dc.sample.zap.filepool.ZapOpenApiScanService).%New().Print("/crud")
```

If you would like to scan all REST APIs in a namespace - USER, for instance, run this command:

```
Do ##class(dc.sample.zap.filepool.ZapOpenApiScanService).%New().PrintAllWebApps("USER
")
```

If yxou suppress the namespace, the current one is used.

Scanner results

This project uses three capabilities of ZAP to provide reports: plain text, HTML and Markdown.

The plain text just shows which tests passed and which failed, as well a summary at the end. A code for details about the OWASP vulnerability is also presented for each test.

```
-----------------------------------------
ZAP API Scan for: /crud
-----------------------------------------
2021-11-29 03:44:16,920 Could not find custom hooks file at /home/zap/.zap_hooks.py
2021-11-29 03:44:32,869 Number of Imported URLs: 8
Total of 19 URLs
PASS: Directory Browsing [0]
PASS: Vulnerable JS Library [10003]
PASS: Cookie No HttpOnly Flag [10010]
...
PASS: Loosely Scoped Cookie [90033]
WARN-NEW: Content Security Policy (CSP) Header Not Set [10038] x 6
        http://host.docker.internal:52773/crud/persons/all (401 Unauthorized)
        http://host.docker.internal:52773/crud/ (401 Unauthorized)
        http://host.docker.internal:52773/crud/_spec (401 Unauthorized)
        http://host.docker.internal:52773/crud/persons/id (401 Unauthorized)
        http://host.docker.internal:52773/crud/persons/id (401 Unauthorized)
...
FAIL-NEW: 0     FAIL-INPROG: 0  WARN-NEW: 3     WARN-
INPROG: 0  INFO: 0 IGNORE: 0       PASS: 74
-----------------------------------------
Markdown: /irisdev/app/zap-pool/report-md/6607713456438727.md
HTML: /irisdev/app/zap-pool/report-html/6607713456438727.html
```

At the bottom of the plain text report, the path for HTML and Markdown reports are presented. These reports are similar and have much more details, like vulnerability description and a quick help on how to fix it, for instance:

Content Security Policy (CSP) Header Not Set

Medium (High)

Description

Content Security Policy (CSP) is an added layer of security that helps to detect and mitigate certain types of attacks, including Cross Site Scripting (XSS) and data injection attacks. These attacks are used for everything from data theft to site defacement or distribution of malware. CSP provides a set of standard HTTP headers that allow website owners to declare approved sources of content that browsers should be allowed to load on that page — covered types are JavaScript, CSS, HTML frames, fonts, images and embeddable objects such as Java applets, ActiveX, audio and video files.

- URL: http://host.docker.internal:52773/crud/persons/id
  - Method: DELETE
  - Parameter: `
  - Attack: `
  - Evidence: `

...

- URL: http://host.docker.internal:52773/crud/persons/id
  - Method: PUT
  - Parameter: `
  - Attack: `
  - Evidence: `

Instances: 6

Solution

Ensure that your web server, application server, load balancer, etc. is configured to set the Content-Security-Policy header, to achieve optimal browser support: "Content-Security-Policy" for Chrome 25+, Firefox 23+ and Safari 7+, "X-Content-Security-Policy" for Firefox 4.0+ and Internet Explorer 10+, and "X-WebKit-CSP" for Chrome 14+ and Safari 6+.

How does it work?

ZAP has several ways to perform security tests, such as scripts or API. This project used execution by scripts executed in an official docker image of ZAP.

So, in order to let the IRIS container execute scripts in the ZAP container, a shared volume was set up in docker-compose.yaml file. In this volume, IRIS writes scripts which are detected and executed by the ZAP container. In the same way, the ZAP container writes out the output in the same shared volume, so the IRIS container can read them.

As an improvement in this project, I ' m planning to use the ZAP API in place of file sharing. An API for executing tests and presenting reports directly in the browser is also planned.

Ideas for using ZAP

We can use it as shown in the example above. However, one idea we had was the possibility of having this integrated within the administration portal.

So, you could select which REST application would pass vulnerability and security testing or create a scheduled task to run once in a while.

The generated report is simple, direct, objective, and well illustrative.

There is also a more detailed version containing, in addition to a description of the vulnerabilities found, tips on how to solve the security problem.

Acknowledgment

Once again, we would like to thank you for all the support from the community in each of the applications we create.

If you found our app interesting and somehow contributed some insight, consider voting for our app.

If you like the app, enjoy what we are doing in the community, please vote for **zap-API-scan-sample** and help us on this journey!

#InterSystems IRIS
Check the related application on InterSystems Open Exchange
