

University of Plymouth

School of Engineering,
Computing, and Mathematics

COMP3000HK

Computing Project

2023/2024

OWASP Penetration Testing Kit:
Enhanced

Kwok Wing Hong

10750763

BSc (Hons) Cyber Security (7142)

Acknowledgements

First, my greatest gratitude to the people of the internet, who shared their knowledge, experience, resources and hard work for everyone, for free. Without their help, I would not have been able to reach this far in this programme, in my lifelong study.

Second, my thanks to the development team of Sublime Text, for developing and continuously supporting my favorite text editor, and making it available for everyone for free. I have been using Sublime Text to write code well before I enrolled in the University of Plymouth, all of my original code in this project. I even use it to take notes for my study and daily life.

Finally, my thanks to my project supervisor, Beta Yip, for his suggestion of a forked project, as well as his guidance and support.

Abstract

OWASP Penetration Testing Kit: Enhanced (OWASP PTK: Enhanced) is a browser extension based penetration tool. A forked project that is built on top of another existing free and open-source software named OWASP Penetration Testing Kit. This project aims to develop an enhanced version of the original software by incorporating new features, as well as exploring the unique advantages browser extension based penetration tools have over penetration tools in traditional standalone software form.

Table of Contents

Acknowledgements	2
Abstract	2
Table of Contents	3
1, Introduction	5
2, Background, Objectives and Analysis	6
2.1, Motive	6
2.2, Objectives	6
2.3, Analysis	7
2.4, Features of OWASP PTK	8
3, Literature and Technology Review - 1500	10
3.1, OWASP Top 10	10
3.2, Zed Attack Proxy (ZAP)	10
3.3, Burp Suite	11
4, Project Management	13
4.1, Project Vision	13
4.2, Method of Approach: Agile methodology	13
4.3, Version Control	13
4.4, Sprint Plan	14
4.5, Risk Management Plans	14
5, Implementation	17
5.1, Scope definition	17
5.2, System Design Approach	17
5.3, Feature 1: Request interceptor	18
5.4, Feature 2: Directory Buster (Dirbuster)	21
5.5, Feature 3: Web Crawler (Spider)	24
5.6, Feature 4: User Agent Switcher	26
5.7, Feature 5: Fuzzer	28
6, Usability Test	31
6.1, Testing Strategy	31
6.2, Evaluation of Interceptor	31
6.3, Evaluation of Directory Buster	33
6.4, Evaluation of Web Crawler	35
6.5, Evaluation of User Agent Switcher	37
6.6, Evaluation of Fuzzer	38
7, Legal, Social, Ethical, And Professional Issues(LSEP)	39
7.1, Legal	39

7.2, Social	40
7.3, Ethical	40
7.4, Professional	41
8, End-Project Review and reflection	42
8.1, Best Aspects	42
8.2, Worst Aspects	42
8.3, Discarded Functions and Features	42
9, Conclusion and Future Works	45
Reference	46

Word count: 9499

Link to code (invite only):

<https://github.com/hkuspace-pu/OWASP-Penetration-Testing-Kit-Enhanced>

1, Introduction

This report outlines the development process of OWASP Penetration Testing Kit: Enhanced (OWASP PTK: Enhanced), a browser extension based penetration testing tool designed for testing the overall security of web applications. OWASP PTK: Enhanced is a forked project that is built on top of an existing free and open-source software named OWASP Penetration Testing Kit. OWASP PTK: Enhanced aims to enhance its penetration testing capabilities by introducing several key new features and functionalities. This report will provide a detailed overview of the development process, highlights of the key features introduced and results of testing and evaluation on the effectiveness of the newly introduced features.

Penetration testing tools are softwares designed for assisting in conducting penetration tests, a practice of simulated non harmful cyber attack against a computer system, network, or web application to proactively identify and patch potential security vulnerabilities before cyberattackers can exploit them. (Shah and Mehtre, 2014) The results of the penetration test provide valuable insights into assessing and improving the overall security strength of the system.

2, Background, Objectives and Analysis

2.1, Motive

In our contemporary world, web applications have emerged as one of the primary means of providing service. A web application is a software that is accessed and interacted with through a web browser, utilizing the Hypertext Transfer Protocol (HTTP), HyperText Markup Language (HTML) and JavaScript. This type of application is designed to provide a user-friendly interface. Enabling greater accessibility in comparison to other kinds of softwares, such as softwares that requires a complex installation process or command line softwares that are not user-friendly.(Alcorn, Frichot and Orru, 2014) Most web applications utilize a client-server architecture to store data, profile the users and process their requests. However, cyberattackers may abuse the client-server architecture in order to sabotage the web application service provider, access data of other users or perform other kinds of malicious activities. Thus implementing comprehensive security measures is necessary for web application service providers to protect their application from being misused by cyberattackers.

Conducting penetration testing is a crucial part in establishing any comprehensive security strategy. It enables web application service providers to proactively identify and patch potential security vulnerabilities before cyberattackers can exploit them by simulating attacks in a non harmful way. Performing penetration tests requires specialized tools in order to increase efficiency or perform complex simulated cyber attacks. The plethora of penetration testing tools available on the internet exhibits a remarkable diversity in their areas of specialization. These testing tools differ in their focus, specializing in various types of attacks, environments, and targets. Some testing tools offer distinctive features that set them apart from others.

The OWASP Penetration Testing Kit browser extension is an open source penetration testing tool designed for web applications, particularly in exploiting the OWASP top 10 vulnerability in web applications. (OWASP, no date) Since its initial release, the testing kit has undergone numerous updates and improvements. However, while still effective, the existing testing kit has limitations in terms of its feature set and functionality.

2.2, Objectives

The primary objective of this project is to design and develop a forked and improved version of the OWASP Penetration Testing Kit (OWASP PTK), developed by Denis Podgurskii, by introducing new features to enhance its functionality, usability, and effectiveness. This improved version will feature functionalities determined by analyzing its existing features and comparing it to alternative penetration testing tools. It will

provide a more comprehensive and wider feature set for penetration testers, enabling it to support more advanced testing scenarios and better stay on par with the alternatives. This report outlines the development process, features, and improvements made to the improved version, as well as an evaluation and analysis of the original and improved version.

2.3, Analysis

One of the most distinct features of OWASP PTK is being a browser extension. A browser extension is a software module that introduces functionality to augment web browsers. Browser extensions can enhance user experience with capabilities beyond the original web browser or assist the users in performing certain tasks with the web browser. (Mozilla, no date) As most penetration testing tools were built as a standalone software, there are numerous advantages as well as disadvantages for penetration testing tools built as a browser extension compared to penetration testing tools that were built as a standalone software.

Similar to web applications, browser extensions also utilizes the HTML and JavaScript to provide a user-friendly interface and greater accessibility. Browser extension requires no need for additional installation process, the users can install browser extensions they have found in a browser extension store with one click. Or they can load the browser extension as a temporary browser extension with a few easy steps.

Second, since browser extensions operate within the web browser environment, they can easily access and interact with web applications, enabling real-time and rapid testing. Testers can quickly access the domains and urls they need to target to perform various tests, such as intercepting and modifying HTTP requests, analyzing responses, and identifying vulnerabilities without the need for extensive setup or configuration.

Third, many browser extensions are designed to work across multiple platforms and devices, enabling it to support an even wider range of testing scenarios. Also, browser extensions can be updated automatically by the web browser, ensuring the users have access to the latest features and security patches without manual intervention. This is especially important for applications that are designed to work across multiple platforms since the method of updating may differ among different devices, causing difficulty for the users.

As for disadvantages, penetration testing tools that were built as a standalone software have better scalability as they are not dependent on the functionality of the web browser. Additionally, standalone softwares deprives their own processing power rather than from the web browser, which enables them to utilize more processing power from devices and use them more efficiently. As a result, standalone software often offers

more comprehensive and complex features and capabilities tailored to specific tasks, and greater customizability to support more testing scenarios compared to browser extensions and identifying vulnerabilities without the need for extensive setup or configuration.

To summarize, penetration testing tools built as a browser extension are generally smaller in scope but offer better accessibility, rapid testing and better cross platform support, in exchange of less comprehensive feature sets, capabilities and customizability. With this in mind, the objective of this project is not to develop an improved version that rivals penetration testing tools built as a standalone software in terms of capabilities, but to preserve its advantages for being built as a browser extension while widening the feature sets. Therefore this goal project would be to develop a forked version of the original PTK.

2.4, Features of OWASP PTK

The original OWASP PTK is a robust penetration testing tool. Featuring numerous functionalities. Below is a list of some of the prominent features of the original OWASP PTK. (OWASP, no date)

- Wappalyzer: A third party tool integrated to the original OWASP PTK. The Wappalyzer provides detailed insight on the technology used by the target web application, such as programming languages, frameworks and their versions, web servers, firewall and third-party libraries. (Wappalyzer, no date)
- Cookie manager: Cookies are small pieces of data sent by the web application server to the local storage of the web browser of the user. It is necessary for many important functions such as authentication, temporary data storage and tracking the browsing activity of the user. (Mozilla, no date) The cookie manager provides a list of cookies collected by the browser. An editor that can modify the content of the cookie. And management options for cookies including delete, create, import, export and blocking specific cookies. Due to the benefit of being a browser extension, it can access and edit cookies in real time.
- Proxy: The proxy is a tool that logs all requests sent to the application from the browser tab with filtering support, enabling the penetration testers to review all requests to conduct comprehensive assessments regarding software mechanisms of the web application. And any request logged can be sent to the R-Builder.
- R-Builder: The R-Builder is a tool that enables the penetration testers to create or modify HTTP/S requests in plain text. As well as logging the response of the

request from the target web application. It can be used to test vulnerabilities such as broken access control and cross-site scripting

- R-Attacker: The R-Attacker is an active vulnerabilities scanner that uncovers vulnerabilities of the target web application by sending unsafe requests automatically and performing analysis on the responses.
- Software composition analysis(SCA): A third party tool based on Retire.JS integrated to the original OWASP PTK. SCA is a scanner for vulnerable JavaScript libraries.

While this is not an exhaustive list of all the features of the original OWASP PTK, it lacks some of the functionalities that are featured in other penetration testing tools. Which will be discussed in detail in the next section.

3, Literature and Technology Review - 1500

The section will identify and examine recently published literature and technology. Providing insights to the current landscape of technology and security threats. Enabling the development of this project to better adapt to the present landscape of technology and security threats and achieve higher effectiveness in fulfilling its objective.

3.1, OWASP Top 10

The Open Worldwide Application Security Project (OWASP) is a non-profit organization and open community that aims to improve the security of software worldwide, by providing freely available articles, documentation, methodologies and tools. (OWASP, no date) The most renowned publication of OWASP is the OWASP Top 10. OWASP Top 10 is a documentation of the top 10 web application security risks worldwide, complete with guidelines to how to prevent or mitigate the risk. The documentation is updated every few years to adapt to the ever changing landscape of technology and security threats. Since the first publication of the documentation in 2003, the documentation has become an industry standard for web application security, recognized by many organizations including The European Union's Agency for Cybersecurity (ENISA). (ENISA, no date)

As the name suggests, OWASP PTK is one of the many tools developed by the OWASP community. It shares the same goal as the OWASP of improving the security of software worldwide. Because of the significance of the OWASP Top 10 and respecting the goal of the original OWASP PTK, the development of OWASP PTK: Enhanced will heavily emphasize on developing new features that assist in uncovering OWASP Top 10 risks of the web application. The OWASP Top 10 will be used as an indicator for assessing the effectiveness of the new features introduced in OWASP PTK: Enhanced.

3.2, Zed Attack Proxy (ZAP)

Formerly known as OWASP ZAP, the Zed Attack Proxy is one of the many tools developed by the OWASP community. (OWASP, no date) It is one of the tools that has been given the flagship project of the OWASP community and is very popular among the community. ZAP is a web application penetration testing tool, built as a standalone software written in Java. Similar to OWASP PTK, ZAP is also a penetration testing tool. Compared to OWASP PTK, ZAP was developed with the following prominent features that OWASP PTK lacks.

- Intercept proxy : One of the most prominent features of ZAP, the intercept proxy redirects traffic sent by the web browsers to itself, enabling the content and traffic to be examined and modified before the traffic is sent to the web application.

- Web crawler: Web crawler is a tool that automatically visits web pages and identifies the URLs it contains. It can recursively visit the URLs extracted from the previous scan and perform the scan again on it, forming a repository of web pages of the target web application. It can assist the penetration testers in mapping the attack surface for further actions and discover hidden URLs.
- Forced browsing: Also known as directory busting, forced browsing is a brute forcing tool that aims to discover unlinked content in the domain directory, such as temporary files or hidden admin control panel unprotected by access control.
- Fuzzer: Fuzzer is a tool that identifies input elements in web applications and automatically fills them with random, unexpected or invalid data. It may cause errors in web applications that did not account for unexpected or invalid data input. It can also be used to simply speed up the penetration testing process by instantly filling all the input elements mandated by the web application.
- Scripting support: Zap enables the penetration testers to import their own custom scripts to interact with the program modules in specific ways. Enabling personalization rule sets and automation.

3.3, Burp Suite

Burp Suite is another popular web application penetration testing tool, built as a standalone software similar to ZAP. (PortSwigger, no date) It features a professional edition that requires the users to purchase a yearly subscription of commercial license, but also provides community edition with reduced functionality for free. As OWASP PTK: Enhanced was designed to be free open source software, the comparison to Burp Suite is based on its free community edition. Compared to both OWASP PTK and ZAP, Burp Suite community edition was developed with the following prominent features that both lacked.

- Repeater: One of the most prominent features of Burp Suite, it enables penetration testers to send web requests or websocket messages in bulk with varying parameter values to test for input-based vulnerabilities. It can also be used as a very effective brute forcing tool in identifying hidden directory, resource, default or weak credentials. It can assess the overall strength of the access control mechanism of the target web application.
- Sequencer: Sequencer is a tool that analyzes the quality of randomness and cryptography of session tokens.
- Clickbandit: Clickbandit is a tool for generating clickjacking attacks. Clickjacking is an attack that tricks the users into clicking a webpage element that is invisible,

hidden under or disguised as another element. (PortSwigger, no date) This can cause the users to unknowingly perform actions against their intention, such as downloading malware, visiting malicious web pages or providing credentials. Clickbandit enables penetration testers to create an attack to confirm that this vulnerability can be successfully exploited.

4, Project Management

4.1, Project Vision

OWASP Penetration Testing Kit: Enhanced (OWASP PTK: Enhanced) is a Firefox browser extension for performing penetration tests against web applications. A forked version of the original OWASP Penetration Testing Kit, featuring additional features and functionalities. Enhancing the efficiency for penetration testers even further along with the original functionalities from OWASP Penetration Testing Kit.

In software development, a fork is a copy of a project that is created by a different team of developers from the original project. Using the entire or part of the source code original project as foundation and start independent development on it, make their own modification or improvement on it. Forks are considered a distinct and a separate piece of software. (Robles and Jes'us, 2012)

4.2, Method of Approach: Agile methodology

In the development of OWASP PTK: Enhanced, the agile methodology was chosen for its adaptability, responsiveness, and iterative development cycles. The defining feature of agile methodology is the iterative development cycles, characterized by incremental changes, refinement, testing and feedback gathering. (Atlassian, no date) As the scope, functional requirements and method of implementation may change after every supervisor meeting and feedback may be provided, this approach enabled the project to effectively navigate the dynamic and ever-changing project environment. The flexibility of the agile methodology enabled for the efficient management of project scope, schedule, and resource. Ensuring the seamless integration of changes and improvements into the project workflow and overall success of this project

4.3, Version Control

Version control is an important aspect of modern software development. From large-scale projects involving many team members to small solo projects. Version control is a practice of tracking and managing changes to the project over time, primarily source code. It enables the retrieval of part of the codebase of previous versions and the reversal of potentially detrimental changes. In this project, GitHub was selected as the version control tool due to its robust features and widespread adoption in the development community. (Atlassian, no date)

Throughout the development process of this project, a version was created for each successfully implemented feature or any significant change to the codebase. As this

project is a solo project, there will always be only one person working on the same files at a time. Thus branching was not necessary for version control, all changes were sent directly to the main branch, preventing any potential merge conflict. Should a need for reversing changes to the codebase arise, the version before the changes can be chosen to revert to.

4.4, Sprint Plan

As Agile methodology was chosen as the method of approach for this project, the software development process of this project was segmented into a series of phases, referred to as sprints. Each sprint was dedicated to a single iterative development cycle, which involves reviewing the project plan, implementing the next major task, testing and gathering feedback from my supervisor. Upon the completion of each sprint, feedback gathered from my supervisor was used for reviewing the project plan and the cycle restarted.

The development of this project took place from December 2023 to May 2024. There are in total 6 sprints, each sprint took a whole month to complete. Here are the sprint plans made for each month of the development process and its associated major task.

- December 2023: Project initiation
- January 2024: Study on software architecture and features of the original OWASP PTK
- February 2024: Implement the interceptor feature
- March 2024: Implement the directory buster and web crawler features
- April 2024: Implement the user agent switcher and form fuzzer features
- May 2024: Final touch-up and testing

4.5, Risk Management Plans

As part of the software development process, it is essential to identify, assess, and mitigate potential risks that could undermine success of the project. Risks to the development of this project along with their respective management strategy implemented to minimize their likelihood and impact has been identified in the table below.

Risk	Management Strategy
------	---------------------

Schedule Overrun/Unexpected delay	Major tasks for each sprint are planned to take 2 to 3 weeks to complete, allowing room for small amounts of unexpected delay. If the delay exceeds the allowance, adjustment on the project plan may be made to account for the delay in the project plan review phase of the next sprint. In the event of a severe setback, if the event can be proven, I may attempt to request an extension period for the project.
Feature Creeping	Inquire my supervisor for advice on the viability and importance of the planned features. If any of the planned features may cause too much delay to the schedule or incompatibility issue. A workaround for the feature may be devised if the schedule allows the delay caused by devising the workaround. If no viable workaround can be devised within a reasonable amount of time, the feature may be discarded.
Difficulties in learning the technologies	The technologies required for the development of the project proved to be too time consuming to learn on my own. I may start working on the major task of the next sprint plan instead and wait for the next meeting with my supervisor to inquire about it. If necessary, features that cannot be completed within a reasonable amount of time may be discarded.
Project data loss	Utilizing the Github repository as a means to backup the project file. In case of a data loss, the project file can be cloned from the repository to restore the progress. However, progress made between the last commit and the event of data loss would still be lost. Thus changes would be committed frequently to minimize the progress loss.

Hardware failure	<p>I would try to buy a replacement part if my small budget can afford it, this failure may cause a few days of unexpected delay to the schedule. Alternatively I can travel to the campus to use campus computers for the development of the project. However, it may cause too much delay as well as accumulating an expensive sum of travel fare.</p>
------------------	--

5, Implementation

5.1, Scope definition

The following considerations were established in order to determine which features and functionalities should be introduced in OWASP PTK: Enhanced.

- Relevance to OWASP Top 10: As previously mentioned in the literature and technology review section, OWASP PTK: Enhanced will heavily emphasize on developing new features that assist penetration testers in uncovering OWASP Top 10 risks of the web application.
- Technological Feasibility: Features that could be implemented within a reasonable amount of time, including the time requirement to study the technology necessary for the implementation.
- Alignment with the original OWASP PTK: Features that complement or synergize with the existing features of the original OWASP PTK. Without compromising the the distinct advantages browser extensions have over standalone software
- Independence: Features that do not rely on third party service providers to function. As external dependency may cause numerous difficulties to the development and stability of the software such as security risk to the users, risks of unavailability of service, difficulty in testing and limited customizability.

After careful consideration and evaluation, the following features were selected for introduction with OWASP PTK: Enhanced.

- Request interceptor
- Directory buster
- Web crawler
- User agent switcher
- Fuzzer

5.2, System Design Approach

As OWASP PTK: Enhanced and the original OWASP PTK were built as a browser extension, for compatibility concern and ease of development, HTML and JavaScript was used as the programming language for OWASP PTK: Enhanced. HTML and JavaScript is adopted by the supermajority of websites as the programming language

for the webpage behavior on the client side. As browser extensions work similarly to a webpage, the supermajority of browser extensions also adopted HTML and JavaScript as the programming language for the client side behavior. (Mozilla, no date)


The original OWASP PTK adopts the model–view–controller (MVC) software design pattern, a common approach to handling interactions between user interface and functionality of the application. The MVC design pattern separates an application into three interconnected components, model, view and controller. (Mozilla, no date)

- Model: The model encapsulates the functionality and background operation part of the application. It is responsible for executing the core functionality of the application by managing data, performing calculations and logic operations.
- View: The view encapsulates the user interface part of the application. It is responsible for presenting the user with information output and prompt the user to provide their input.
- Controller: The controller is a module that handles the interaction between the model and the view. It receives input by the user from the view, and passes them to the model, and then passes the output from the model back to the view.



The MVC pattern encourages the practice of separation of concerns, each component should have only one specific responsibility, making it easier to maintain, understand and update the code. Enabling a team of developers to work on different parts of the application independently with minimal conflict. For compatibility concerns and ease of development, the MVC design pattern was also adopted for implementing the additional features and functionalities of OWASP PTK: Enhanced.





5.3, Feature 1: Request interceptor

Featured in both ZAP and Burp Suite, the interceptor is one of the most prominent features of both aforementioned software and the most complex feature introduced in OWASP PTK: Enhanced. Much like the functionality of the interceptor featured in both ZAP and Burp Suite, it enables the outward web requests to be intercepted, examined and modified before the web request is sent to the web application server.


Dashboard Cookies SCA Proxy R-Builder R-Attacker JWT Inspector Decoder Macro Tools

Interceptor DirBuster Spider Agent Switcher Fuzzer

<https://juice-shop.herokuapp.com/#!/score-board>
On/Off ☒



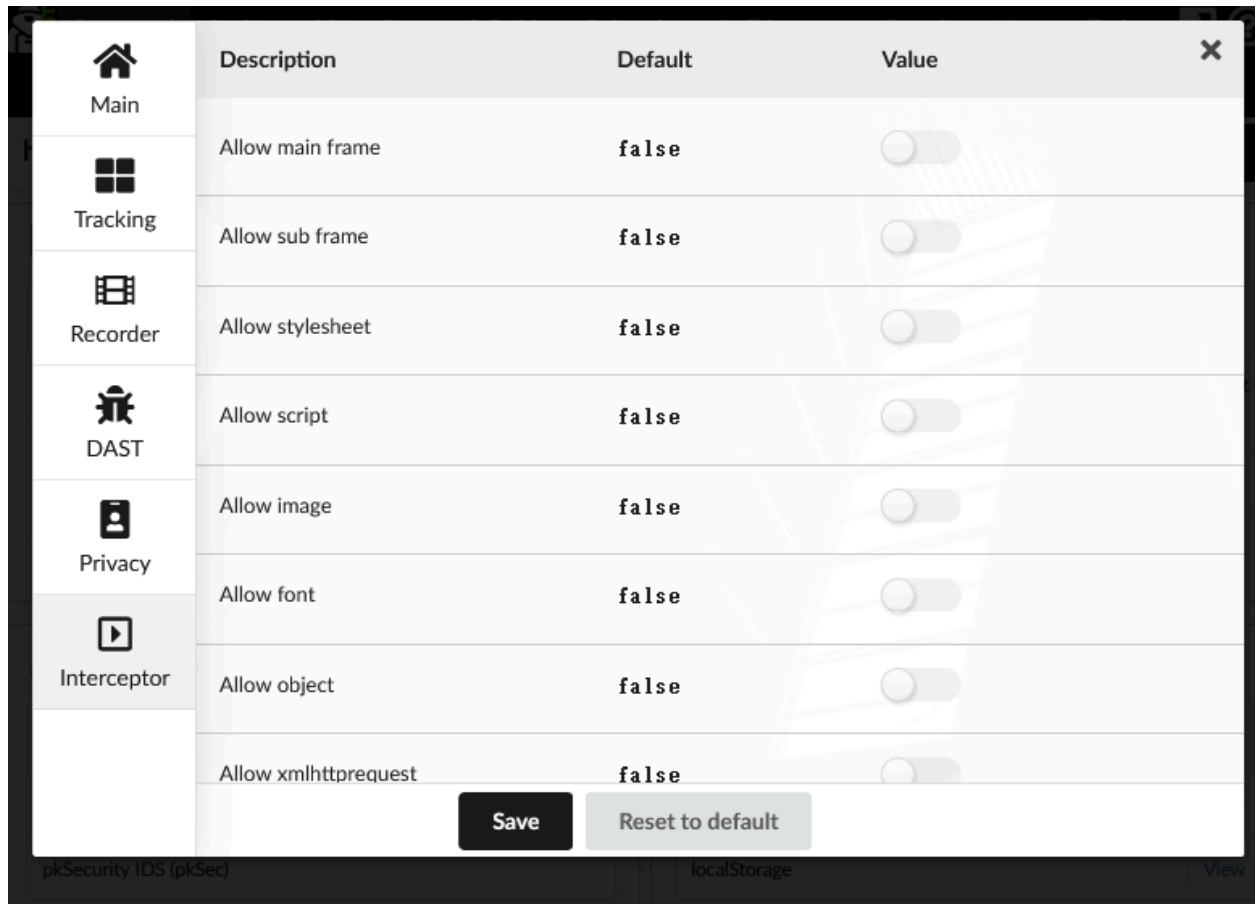
Send to R-builder	Details	URL	Request ID	Method	Type
		https://juice-shop.herokuapp.com/rest/products/search?q=	28446	GET	xmlhttprequ
		https://juice-shop.herokuapp.com/#!/	28447	GET	main_frame

```

requestId:28447
url:https://juice-shop.herokuapp.com/#!/
method:GET
type:main_frame
timeStamp:1716933942800
tabId:100
frameId:0
parentFrameId:-1
incognito:true
thirdParty:false
cookieStoreId:firefox-private
proxyInfo:null
ip:null
frameAncestors:[]
urlClassification:thirdParty

```

Accessed through clicking on the interceptor tab in the toolbar at the top, it will provide an interface with a switch that can turn on or off the intercepting function, a button that exports the result into a TXT file and a button that clears the record. Below them is a table that will display the records of all the requests that were successfully intercepted with their destination URL, request ID, request method and request type. Each record has a button that sends the intercepted request to the R-Builder, and a button that expands the row and displays the full content of the request. The headers of the table can be clicked to sort the table in ascending or descending for better user experience.



A new tab has been added to the settings menu that enables penetration testers to specify the type of request they wish to intercept, allowing requests to be selectively intercepted based on the preference of the users. By default, all types of requests will be intercepted.

Implemented with the MVC design pattern, the functional part of the feature is separated from the user interface, enabling the interceptor to stay running even if the user interface is closed.

By intercepting and analyzing the content of outward web requests, it enables penetration testers to conduct comprehensive assessments regarding the software architecture, security mechanisms and cryptographic mechanisms of the web application. Assisting in uncovering many of the OWASP top 10 web application risks.

- Cryptographic failure (A2: 2021): Analysis of the content of outward web requests may show sensitive data being sent in cleartext, weak cryptographic algorithms or protocols, reused cryptographic keys and weak key sizes.
- Identification and Authentication Failures (A7:2021): Analysis of the content of outward web requests may show exposed session identifier in the URL, reuse

sessioned identifier after successful login and insecure handling of session tokens

- Security Misconfiguration (A5: 2021): Analysis of content of outward web requests may show the use of unnecessary features, services or insecure handling of XML Http Requests (XHR).
- Vulnerable and Outdated Components (A6:2021): Analysis of content of outward web requests may show the use of outdated or unpatched libraries and insecure dependencies on third party services.
- Injection (A2:2021): Analysis of the content of outward web requests may show the absence of input sanitization and validation.

Additionally, it complements very well with the proxy and R-Builder features of the original OWASP PTK. As mentioned before, the intercepted requests can be sent to R-Builder and used as a foundation for crafting modified requests to launch further tests. And unlike proxy, the interceptor can capture outward web requests before it reaches the web application server and a trigger response. However, it can only log requests that were intercepted. Proxy on other hands, logs all requests that were not intercepted.

5.4, Feature 2: Directory Buster (Dirbuster)

The Dirbuster is a brute forcing tool that utilizes a wordlist to discover unlinked content in the domain directory.

Dashboard Cookies SCA Proxy R-Builder R-Attacker JWT Inspector Decoder Macro Tools

Interceptor DirBuster Spider Agent Switcher Fuzzer

https://juice-shop.herokuapp.com/#/ Scan Lock Download Delete

Origin URL

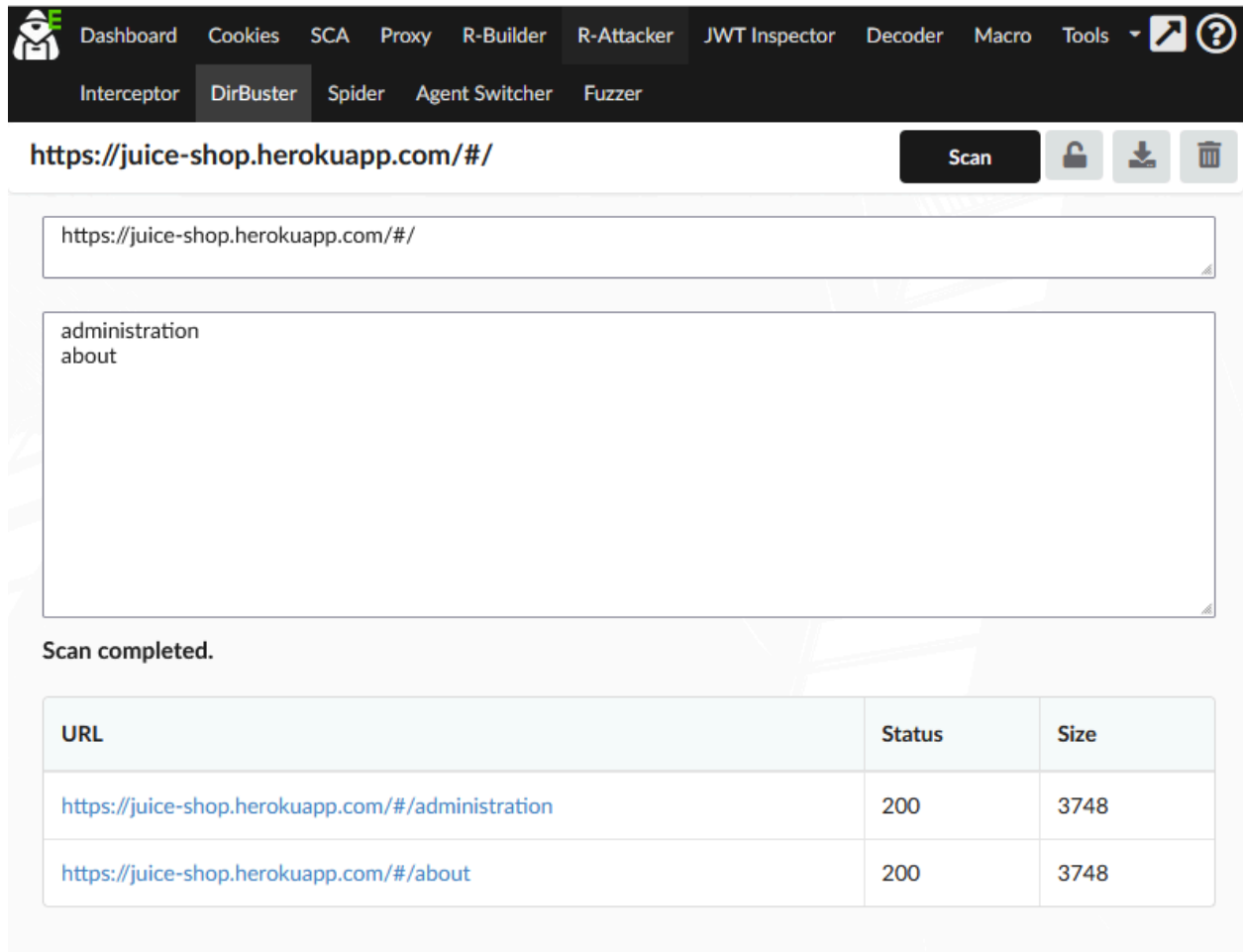
Copy and paste your wordlist here (limit 3000 lines)

Wordlist and scan record cleared.

URL	Status	Size
-----	--------	------

Accessed through clicking on the DirBuster tab in the toolbar at the top, it will provide an interface with a scan button that activates the Dirbuster, a button that locks the origin URL, a button that exports the result into a TXT file and a button that clears the result. Below them are two text input boxes for the origin URL and the wordlist.

The origin URL is always automatically changed to the current browser tab. If the lock origin URL button is pressed, it will not change unless manually changed by the user.



https://juice-shop.herokuapp.com/#/

Scan

administration
about

Scan completed.

URL	Status	Size
https://juice-shop.herokuapp.com/#/administration	200	3748
https://juice-shop.herokuapp.com/#/about	200	3748

When the Dirbuster is activated, it will begin to send requests with destination URL made from combining the origin URL with the a word on the wordlist until the wordlist is exhausted. If the response code of the request was not 404 (not found), it will be recorded and displayed on the table below. The URLs in the table can be clicked and the user will be directed to the displayed URL.

Due to risk of misuse, the wordlist only allows up to 3000 words to be used for testing at a time. As the brute forcing nature of this tool may generate a large influx of traffic. Some web applications may not be able to handle a large influx of traffic, causing unavailability of service.

Like the interceptor, the headers of the table can be clicked to sort the table and the tool will stay running even if the user interface is closed. It is necessary as the scan could take many minutes or even hours to complete.

By scanning for unlinked content with brute forcing method, the result can be used to assist in uncovering many of the OWASP top 10 web application risks.

- Insecure design (A4:2021): Inclusion of unlinked content indicates that redundant code, resources, files are not removed and exposed. Redundant code, resources or files often pose security risks. For example, error message logs may allow cyberattackers to learn of other vulnerabilities of the web application. Code files may allow cyberattackers to learn of other vulnerabilities by reverse engineering the code. Leakage of personal data or credentials of other users. In the worst case, an exposed admin control panel may allow total take over of the web application by cyberattackers.
- Broken access control (A1:2021): Exposed admin control panel or other privileged content indicates that authentication mechanism can be bypassed through forced browsing
- Identification and authentication failures (A7:2021): It should be noted that this tool should not work on properly secured web applications due to its brute forcing nature. If a scan with a large wordlist can be completed without triggering any brute force protection measure indicates the absence of such security measure.
- Security misconfiguration (A5:2021): Redundant and exposed content may indicate the inclusion of other unnecessary features which may potentially contain other vulnerabilities that can be exploited by cyberattackers.

Additionally, it synergizes well with the R-Builder of the original OWASP PTK as R-Builder can be used to launch further tests on the exposed content to discover more vulnerabilities and security risks.

5.5, Feature 3: Web Crawler (Spider)

The Spider is a brute forcing tool that extracts URLs from various pages of the target web application to form a repository of web pages of the target web application. This

tool is in many ways similar to the DirBuster.

The screenshot shows the Spider tool interface. At the top is a dark navigation bar with icons and labels for various tools: Dashboard, Cookies, SCA, Proxy, R-Builder, R-Attacker, JWT Inspector, Decoder, Macro, Tools, and a help icon. Below this is a sub-bar with 'Interceptor', 'DirBuster', 'Spider' (highlighted), 'Agent Switcher', and 'Fuzzer'. The main area has a text input field containing 'http://www.itsecgames.com/' and a 'Scan' button. To the right of the input field are three icons: a lock, a download, and a trash. Below the input field, it says 'Max Depth: 1' with a dropdown arrow. The status 'Scan completed.' is displayed. A table below shows the scan results with columns for URL, Depth, and Size.

URL	Depth	Size
http://www.itsecgames.com	0	3651
http://www.itsecgames.com/bugs.htm	1	3890
http://www.itsecgames.com/download.htm	1	4673
http://www.itsecgames.com/training.htm	1	3505
http://itsecgames.blogspot.com	1	148098
http://www.itsecgames.com/./downloads/vulnerabilities.txt	1	5965
http://www.itsecgames.com/./downloads/bWAPP_intro.pdf	1	4766469

Accessed through clicking on the Spider tab in the toolbar at the top. Similar to the implementation of the DirBuster, it features an interface with a scan button, a button that locks the origin URL, a button that exports the result and a button that clears the result. With a sortable table below that displays the recorded results with clickable URL. The origin URL also works similarly to the implementation of the DirBuster. And lastly, a selector of the max depth that determines the maximum depth of the scan.

When the Spider is activated, the Spider will first visit the web page of the origin URL, identifying and recording all the URLs it contains, marking the depth of the results as 1. If the max depth is greater than one, the Spider will repeat the same scanning process on all the URL results, incrementing the depth level by 1, and marking the depth of the new results with the current depth level. This process repeats until the current depth level has reached the maximum depth.

Due to risk of misuse, the maximum depth can only be set at 5 at most. As the brute forcing nature of this tool may generate a large influx of traffic, similar to the DirBuster.

The scan could take many minutes to complete and the tool will stay running even if the user interface is closed.

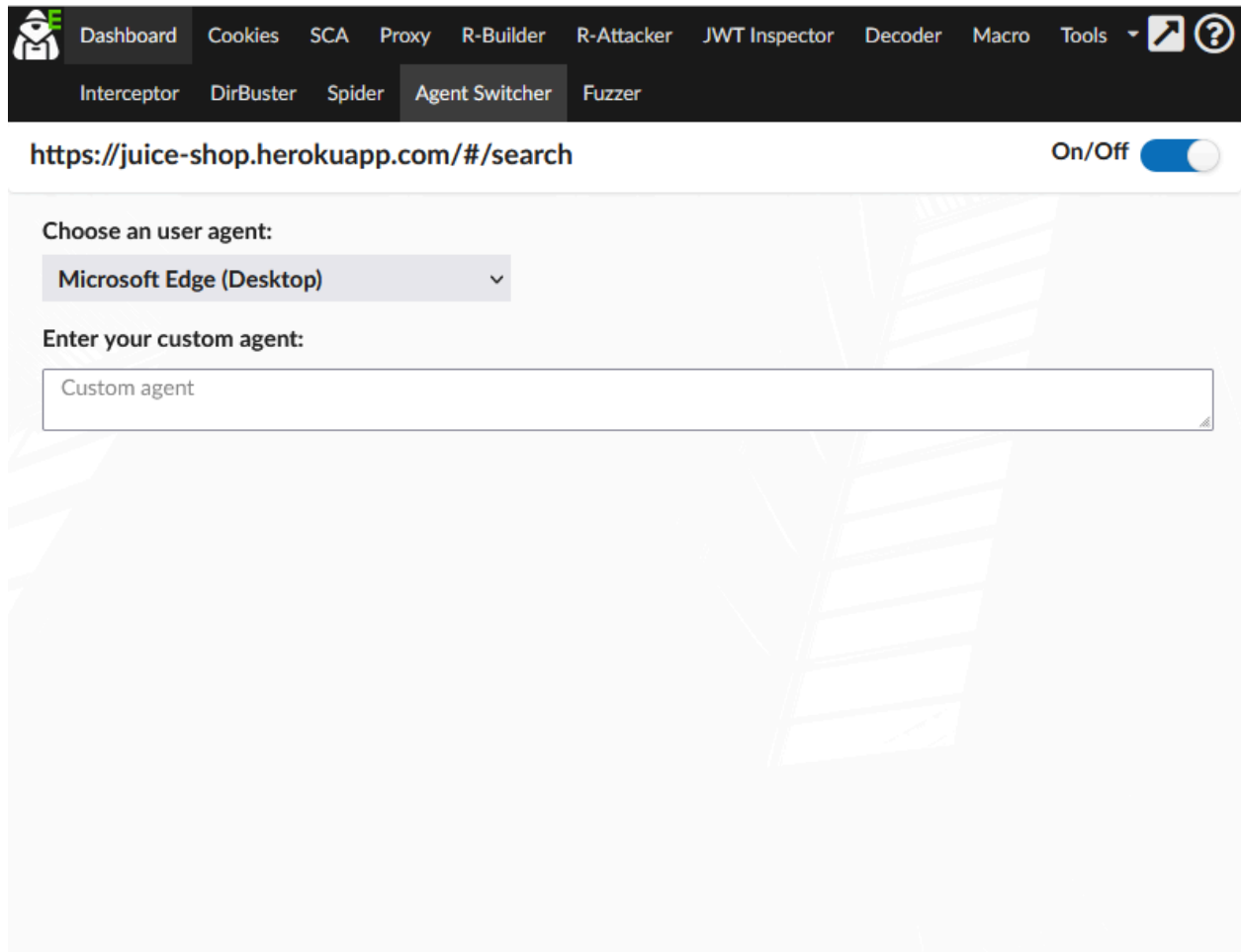
Mapping the URLs of the target web application can uncover potential hidden URLs, which is an indicator of many of the OWASP top 10 web application risks.

- Identification and authentication failures (A7:2021): Similar to DirBuster, this tool should not work on properly secured web applications due to its brute forcing nature. A successful scan with a high depth indicates the absence of brute force protection measures.
- Insecure design (A4:2021): Hidden URLs to redundant and unremoved content may be uncovered. As mentioned in the DirBuster section, redundant content often poses other security risks.
- Broken access control (A1:2021): Hidden URLs to privileged content may be uncovered indicating that authentication mechanism can be bypassed with those hidden URLs or through forced browsing
- Security misconfiguration (A5:2021): Redundant and unremoved hidden URLs may indicate the inclusion of other unnecessary features which may potentially contain other vulnerabilities that can be exploited by cyberattackers.

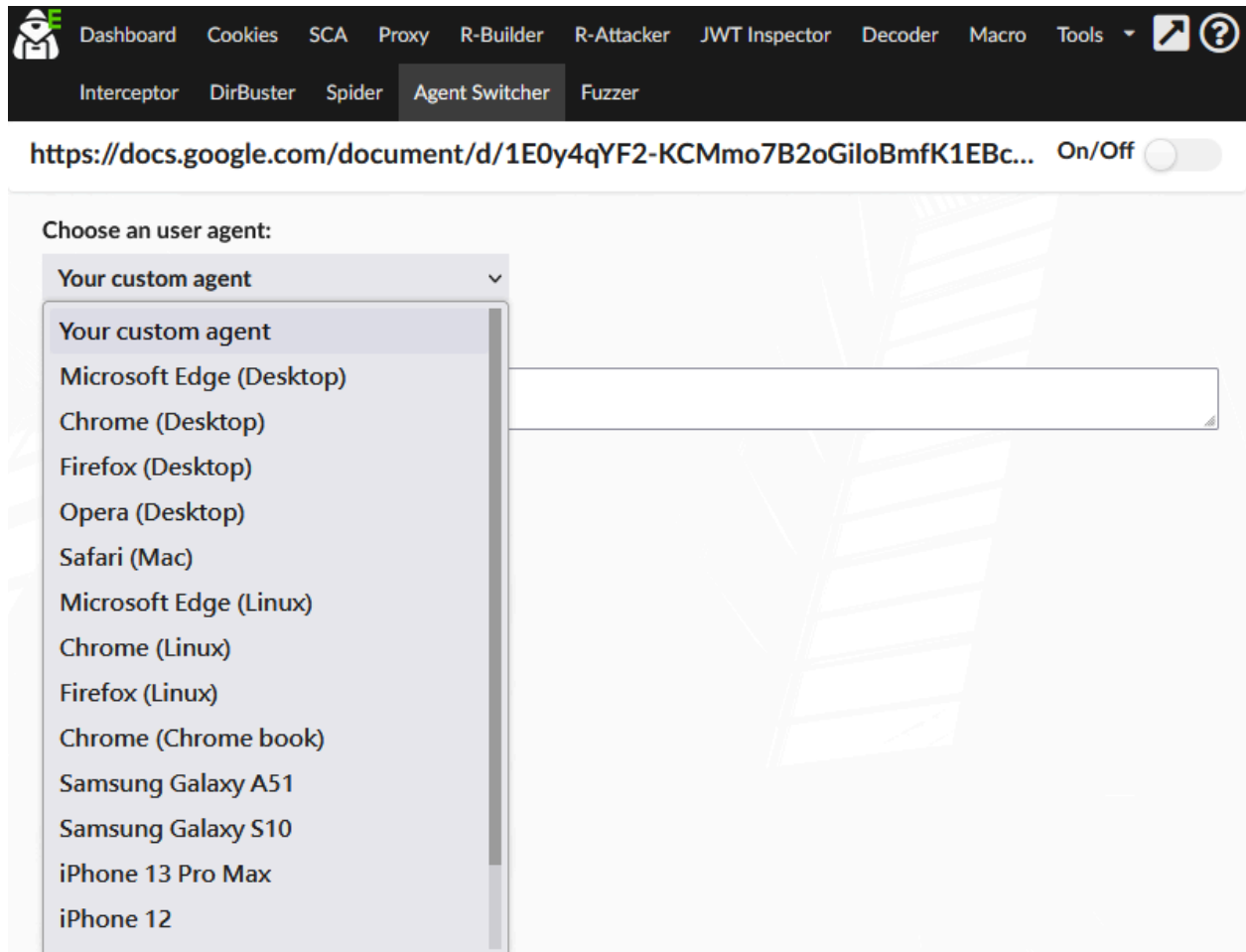
By mapping the URLs of the target web application, a repository of web pages of the target web application can be formed. This repository can assist the penetration tester in mapping the attack surface, understanding the size and architecture of the target web application and identifying hidden elements or interactive elements that may be exploited. Assisting the planning of the whole penetration test.

5.6, Feature 4: User Agent Switcher

The user agent is a string of information that web browsers send to a web application server to identify itself, including information such as the type of web browser, operating system, and device. (Mozilla, no date) Many modern web applications feature more than one way of handling web requests, such as providing mobile phone users a simplified interface. The user agent switcher modifies the user agent in the requests sent to the web applications, enabling penetration testers to test the differences in the ways the target web application handles web requests from different web browsers, operating systems, and devices.



Accessed through clicking on the agent switcher tab in the toolbar at the top. It features an interface with a switch that can turn on or off the agent switching function. A selector for user agent presets and an input for custom user agent.

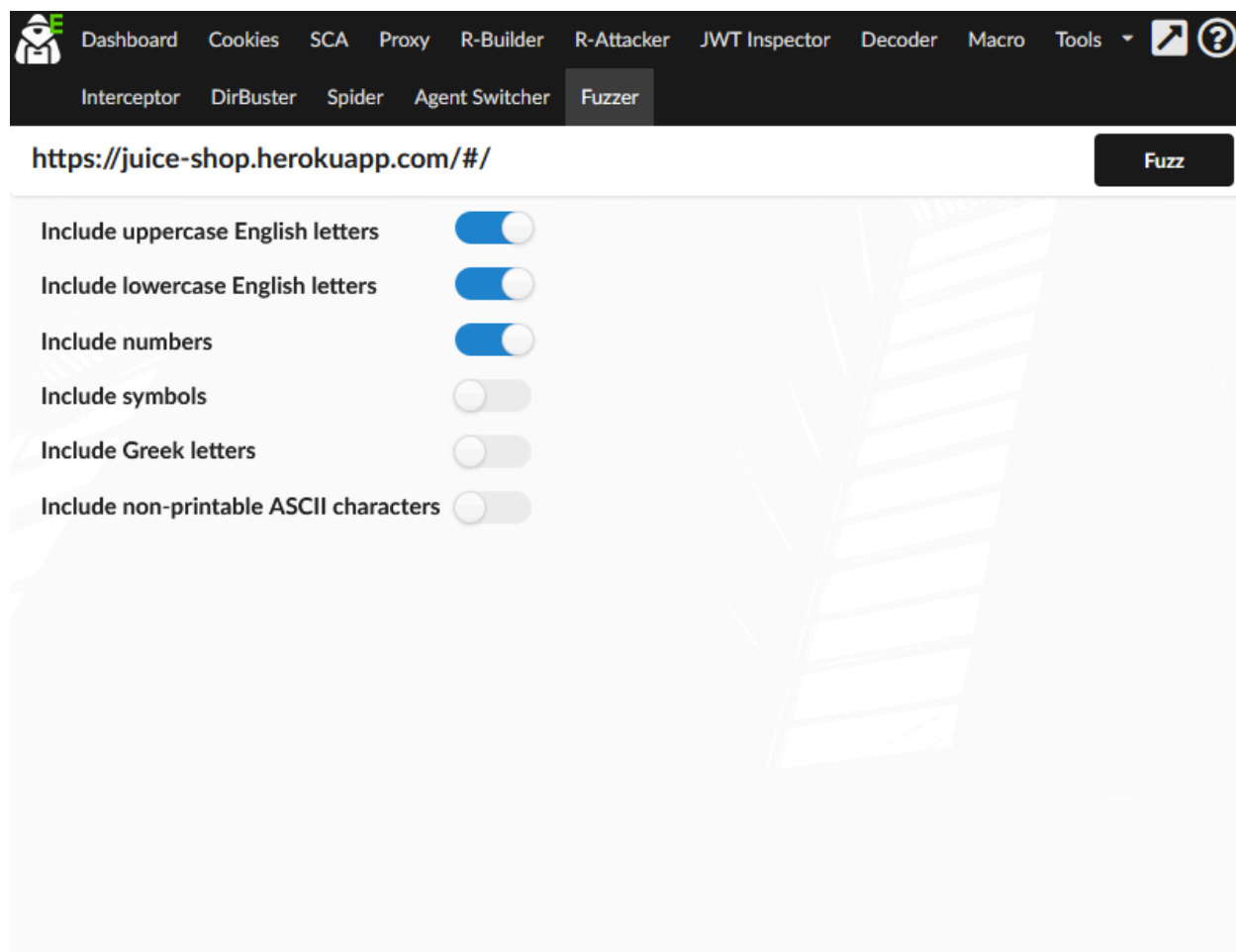


Many user agent presets are included in this tool, differing between the type of web browser, operating system, and device. The users can also enter their own custom user agent string that is missing from the presets.

Although this tool does not directly uncover any of the OWASP top 10 web application risks. It can assist the penetration tester in uncovering vulnerabilities of browser or device specific features or inconsistency in the implementation of different user interfaces. As well as enabling penetration tester to conduct the test with the convenience of using their preferred browser or device

5.7, Feature 5: Fuzzer

The fuzzer is a tool that identifies input elements in web applications and automatically fills them with random, unexpected or invalid data.



Accessed through clicking on the fuzzer tab in the toolbar at the top. It features an interface with a button that executes the fuzzer in the current browser tab. A six switches below that enables the users to customize the random input.

When the fuzzer is executed, it will automatically fill all the input elements of the web page, including hidden input elements, with random strings that contain the enabled character type. The supported character types are uppercase and lowercase English letters, numbers, symbols, Greek letters and non-printable ASCII characters. Numbers, upper case and lower case English letters are enabled by default.

Fuzzer can be used to speed up the penetration testing process by instantly filling all the inputs mandated by the web application. It may also cause errors if the implementation of the web applications did not account for unexpected or invalid data input, or if hidden input elements not meant to be filled in normal circumstances are somehow filled.

Symbols and non-printable ASCII characters are especially prone to cause errors in web applications when processed directly without proper input sanitization and

validation. While Greek letters may usually be handled by modern web applications without major issues, some Greek letters look indistinguishable from English letters and could be used to bypass filters for malicious messages or data exfiltration.

Input sanitization is a security mechanism that ensures input data provided by the users is safe for the software and free from malicious or unexpected data. By identifying and modifying unsafe characters in input data provided by the users before it is processed by the software to prevent error, security vulnerabilities or unexpected behavior. (Shar and Tan, 2012)

By testing the input elements of the target web application with fuzzer, many of the OWASP top 10 web application risks could be uncovered .

- Injection (A2:2021): Improper handling of symbols, non-printable ASCII characters and Greek letters indicates the absence of input sanitization and validation. Which leads to many more potential vulnerabilities.
- Broken access control (A1:2021): Hidden input elements that can be filled with fuzzer indicates that authentication mechanism can be bypassed with tools
- Security misconfiguration (A5:2021): Hidden input elements may indicate the inclusion of other unnecessary features or content.

6, Usability Test

During the development process of OWASP PTK: Enhanced, unit tests of the newly developed features were conducted concurrently with the coding process. This is possible due to the programming language used for browser extension JavaScript, which does not require compilation before execution. Testing and debugging were done by merely applying the changes and reloading the browser extension. As a result, the time requirement for the development of OWASP PTK: Enhanced was drastically reduced.

After the implementation of all the new features of OWASP PTK: Enhanced, a comprehensive test with specialized test strategy will be used to evaluate the quality and effectiveness of the newly introduced features.

6.1, Testing Strategy

To evaluate the effectiveness of the newly introduced features, a specialized test environment is required for each feature. There are many web applications built for security training and testing for security tools. Some of them are hosted on the internet and available for all, others require setting up a virtual machine to host the web application. For the project, online hosted testing web applications will be used for a more realistic environment and reduced time requirement.

The evaluation will be conducted in this format.

- Test environment
- Expected result
- Actions
- Actual result
- Evaluation

6.2, Evaluation of Interceptor

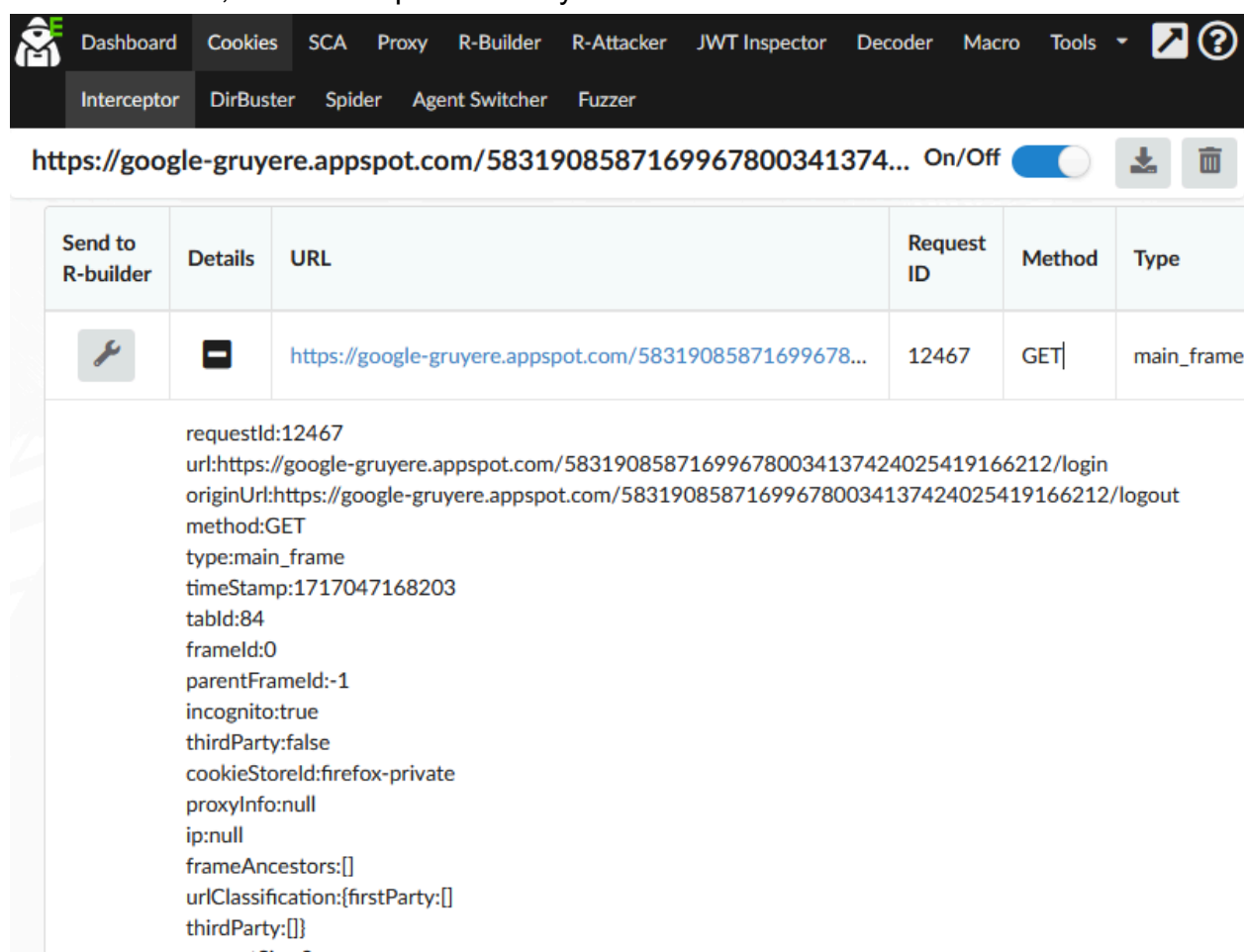
Test environment used: Google Gruyere. A web application created by Google as a training ground for learning web application vulnerabilities. It is purposefully built with various vulnerabilities commonly found in other web applications. The policy of Google Gruyere states authorization to attack the web application is granted to everyone so long as they follow the direction.

Expected results: After the interceptor is turned on, the web browser will not be able to send any web request to the web application server. The intercepted request will be recorded and displayed in the table. Pressing the buttons in the detail column will show the full detail of the intercepted request. Pressing the buttons in the send to R-builder



column will change the interface to the R-builder interface, with the detail of the request carried over.

Actions: Interceptor was turned on, and then attempted to navigate to the login page of the site. A new row appears on the table and the buttons on the table are pressed.

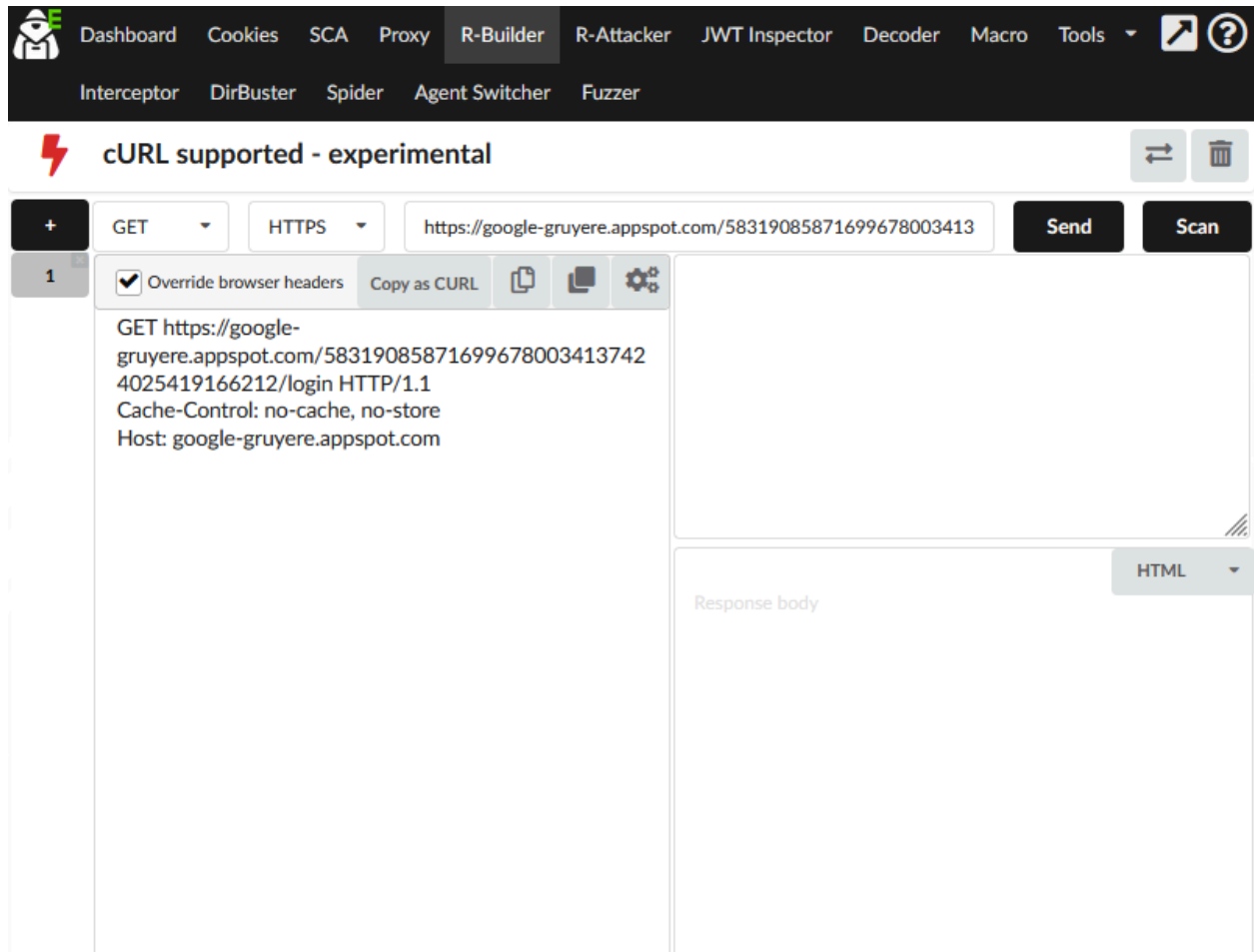
Actual results: After the interceptor is turned on, when attempting to click on the Sign-in hyperlink, no changes happened to the web page. A new row appeared on the result table with the URL that points to the Sign-in page of the web application. Pressing on the he button in the detail column showed the full detail of the intercepted request. Pressing the button in the send to R-builder column changed the popup interface to that of the R-builder, with the request already written.



The screenshot shows the Burp Suite interface. The top navigation bar includes tabs for Dashboard, Cookies, SCA, Proxy, R-Builder, R-Attacker, JWT Inspector, Decoder, Macro, Tools, and a help icon. Below this, a secondary bar contains Interceptor, DirBuster, Spider, Agent Switcher, and Fuzzer. The main window displays the URL <https://google-gruyere.appspot.com/5831908587169967800341374...> with an On/Off toggle and download/delete icons. Below the URL bar is a table with the following columns: Send to R-builder, Details, URL, Request ID, Method, and Type. A single row is present in the table with a wrench icon in the 'Send to R-builder' column, a minus icon in the 'Details' column, the same URL in the 'URL' column, '12467' in the 'Request ID' column, 'GET' in the 'Method' column, and 'main_frame' in the 'Type' column. Below the table, the details of the intercepted request are displayed in a text area.

Send to R-builder	Details	URL	Request ID	Method	Type
		https://google-gruyere.appspot.com/5831908587169967800341374...	12467	GET	main_frame

```
requestId:12467
url:https://google-gruyere.appspot.com/583190858716996780034137424025419166212/login
originUrl:https://google-gruyere.appspot.com/583190858716996780034137424025419166212/logout
method:GET
type:main_frame
timeStamp:1717047168203
tabId:84
frameId:0
parentFrameId:-1
incognito:true
thirdParty:false
cookieStoreId:firefox-private
proxyInfo:null
ip:null
frameAncestors:[]
urlClassification:{firstParty:[]
thirdParty:[]}
```

Evaluation: Actual result perfectly meets the expected. All functional requirements have been met, the primary functions of the interceptor are successfully implemented.

6.3, Evaluation of Directory Buster

Test environment used: OWASP Juice Shop. (OWASP, no date) A simulated juice shop web application purposefully built with many security weaknesses and vulnerabilities, another Flagship project of OWASP. The application is hosted on Heroku, a platform as a service that enables developers to build, run, and operate applications entirely in the cloud. (Heroku, no date)

The policy of OWASP Juice Shop states that permission is granted by default to any person to use OWASP Juice Shop for the purpose of security training, creating awareness demos and as a test subject for security tools.

Expected results: After entering a few words to the wordlist, clicking on the scan button starts the scan. If the the destination URL matched an existing and accessible URL in the web application, a the result should be recorded and displayed in the table

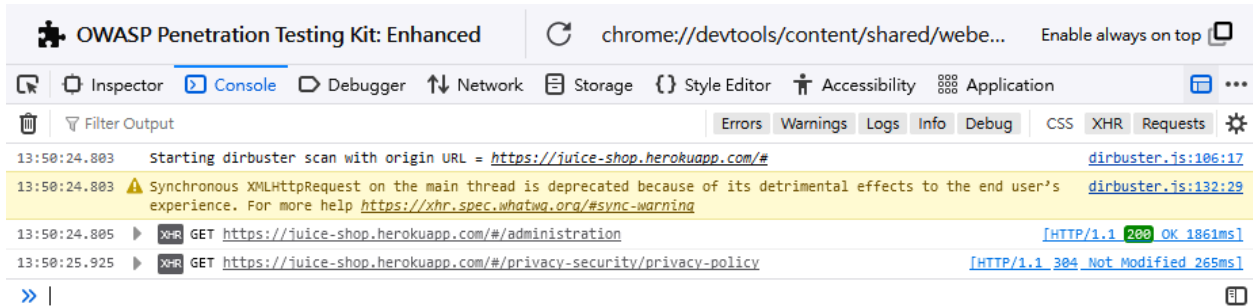
Actions: The word “administration” and “privacy-security/privacy-policy” were entered to the wordlist, then the scan button was pressed.

Actual results: Two new rows appeared in the table with clickable URLs, clicking on the URLs does navigate to two pages not normally accessible by the users.

The screenshot shows the Burp Suite interface with the DirBuster tab selected. The target URL is <https://juice-shop.herokuapp.com/#/>. The wordlist contains the words "administration" and "privacy-security/privacy-policy". The scan has completed, and the results table shows two new URLs discovered.

Scan completed.

URL	Status	Size
https://juice-shop.herokuapp.com/#/administration	200	3748
https://juice-shop.herokuapp.com/#/privacy-security/privacy-policy	200	3748



Evaluation: Actual result perfectly meets the expected. All functional requirements have been met, the primary functions of the interceptor are successfully implemented.

6.4, Evaluation of Web Crawler

Test environment used: Google Gruyere.

Expected results: After pressing the scan button on the login page, the hidden URLs should be uncovered

Actions: After navigating to the login page, the scan button was pressed.

Actual results: A list of URLs appeared in the table, among the list are URLs that are hidden from the users.

https://google-gruyere.appspot.com/58319085871699678003413...

Scan

https://google-gruyere.appspot.com/583190858716996780034137424025419166212/	1	3482
https://google-gruyere.appspot.com/583190858716996780034137424025419166212/login	1	2591
https://google-gruyere.appspot.com/583190858716996780034137424025419166212/newaccount.gtl	1	2961
https://google-gruyere.appspot.com/#	1	11506
https://google-gruyere.appspot.com/583190858716996780034137424025419166212/snippets.gtl?uid=cheddar	1	3379
https://images.google.com/?q=cheddar+cheese	1	197540
https://google-gruyere.appspot.com/583190858716996780034137424025419166212/snippets.gtl?uid=brie	1	3059
https://news.google.com/news/search?q=brie	1	1369798

OWASP Penetration Testing Kit: Enhanced
chrome://devtools/content/shared/webe...
Enable always on top

Inspector
Console
Debugger
Network
Storage
Style Editor
Accessibility
Application

Filter Output
Errors
Warnings
Logs
Info
Debug
CSS
XHR
Requests

14:06:51.970 Starting spider scan with depth = 1 spider.is:100:17
14:06:51.975 GET https://google-gruyere.appspot.com/583190858716996780034137424025419166212 [HTTP/3 302 348ms]
14:06:52.312 GET https://google-gruyere.appspot.com/583190858716996780034137424025419166212/ [HTTP/3 200 323ms]
14:06:52.641 GET https://google-gruyere.appspot.com/583190858716996780034137424025419166212/ [HTTP/3 200 417ms]
14:06:53.068 GET https://google-gruyere.appspot.com/583190858716996780034137424025419166212/login [HTTP/3 200 304ms]
14:06:53.375 GET https://google-gruyere.appspot.com/583190858716996780034137424025419166212/newaccount.gtl [HTTP/3 200 351ms]
14:06:53.730 GET https://google-gruyere.appspot.com/ [HTTP/3 200 198ms]
14:06:53.933 GET https://google-gruyere.appspot.com/583190858716996780034137424025419166212/snippets.gtl?uid=cheddar [HTTP/3 200 358ms]
14:06:54.295 GET https://images.google.com/?q=cheddar+cheese [HTTP/3 200 174ms]
14:06:54.458 Some cookies are misusing the recommended "SameSite" attribute 12
14:06:54.478 GET https://google-gruyere.appspot.com/583190858716996780034137424025419166212/snippets.gtl?uid=brie [HTTP/3 200 356ms]
14:06:54.840 GET https://news.google.com/news/search?q=brie [HTTP/3 302 116ms]
14:06:54.962 GET https://news.google.com/search?q=brie [HTTP/3 302 147ms]
14:06:55.115 GET https://news.google.com/search?q=brie&hl=en-US&gl=US&ceid=US:en [HTTP/3 200 899ms]

Evaluation: Actual result perfectly meets the expected. All functional requirements have been met, the primary functions of the interceptor are successfully implemented.

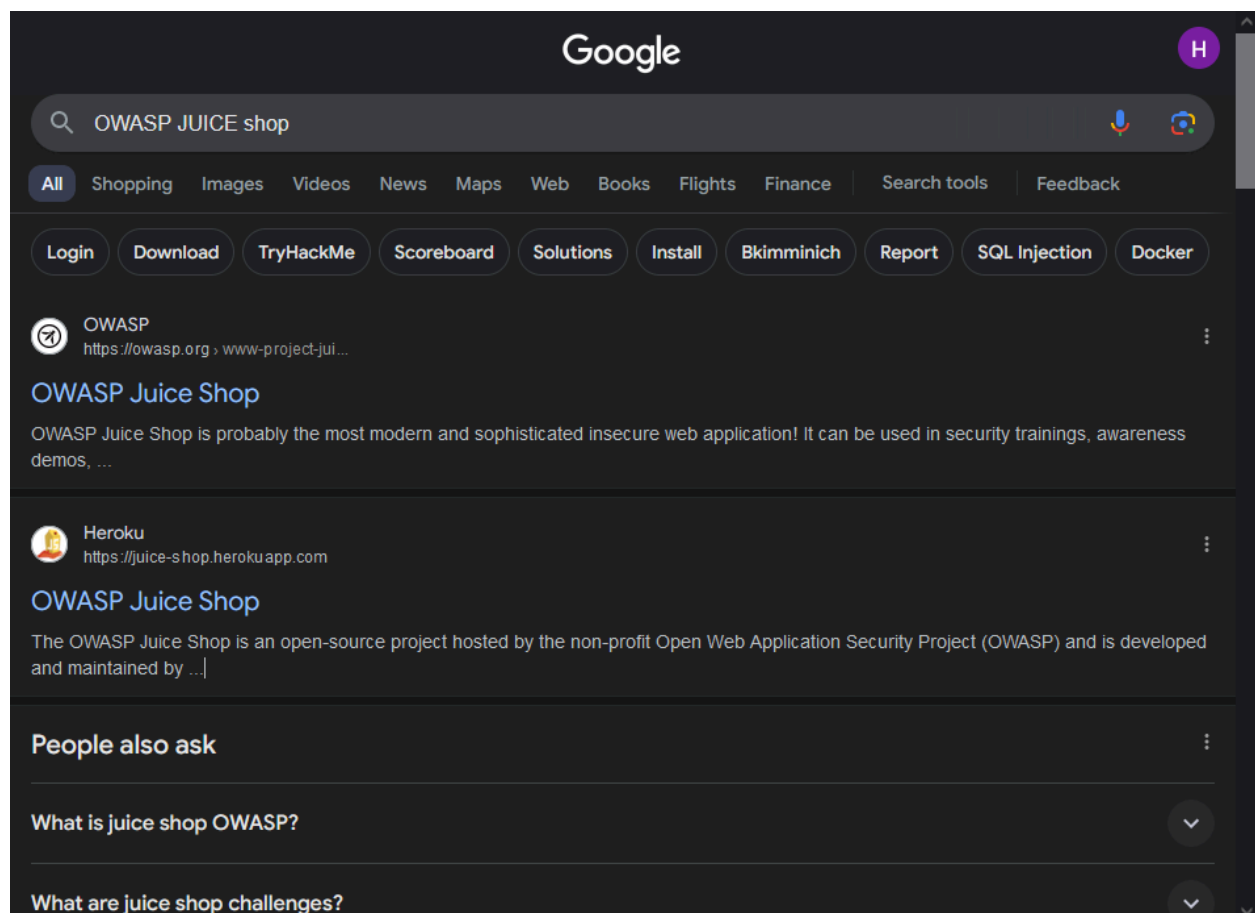
6.5, Evaluation of User Agent Switcher

Test environment used: A Google search result page

Expected results: After selecting the “Samsung Galaxy A51” user agent from the preset, turning on the user agent switcher, and then reloading the page. The mobile interface of the search result page should be shown.

Actions: The “Samsung Galaxy A51” user agent was from the preset, and the user agent switcher was turning on, the web page was then reloaded.

Actual results: The mobile interface of the search result page was shown after reloading.



Evaluation: Actual result perfectly meets the expected. All functional requirements have been met, the primary functions of the interceptor are successfully implemented.


6.6, Evaluation of Fuzzer

Test environment used: The edit profile page of Google Gruyere.

Expected results: After turning on the include symbols option on the interface and the pressing the fuzz button, all input boxes of the web page should be filled instantly with strings consist of English letters, numbers and symbols

Actions: The include symbols option on the interface was turned on and the fuzz button was pressed

Actual results: All input boxes of the web page are instantly filled with strings consist of English letters, numbers and symbols.



The screenshot shows the 'Gruyere: Profile' page. At the top, there is a navigation bar with links: Home | My Snippets | New Snippet | Upload. On the right, it says 'y:aB#-OL1=N>j <123> | Profile | Sign out'. The main heading is 'Gruyere: Profile'. Below it, the section is 'Edit your profile.'.

The form contains the following fields:

- User id: 123
- User name: N*qrGbc bFILXBm>
- OLD Password: [Redacted with dots]
- NEW Password: [Redacted with dots]
- Warning: WARNING: Gruyere is not secure. Do not use a password that you use for any real service.
- Icon: =pAYrDrkTFO<TqV (32x32 image, URL to image location)
- Homepage: An[mJ/P",?DXt~e
- Profile Color: C~~yZBlzT=&U&j
- Private Snippet: XBU DTwZr&'ubH%UH[W@m(,T_EnCt!rmE&!XgC'e%N}]=KCem!Y

Evaluation: Actual result perfectly meets the expected. All functional requirements have been met, the primary functions of the interceptor are successfully implemented.

7, Legal, Social, Ethical, And Professional Issues(LSEP)

As penetration testing tools can very easily be misused for malicious purposes, the LSEP issues of this project were studied with extra care and addressed as comprehensively as possible. To ensure compliance with laws and regulation, mitigate potential societal impacts, uphold ethical standards, and maintain professionalism in the field of cybersecurity and software development .

7.1, Legal

- Intellectual property rights: The original OWASP PTK was published under The MIT License (Open Source Initiative, no date), permission is granted free of charge to any person who has acquired this software to freely use, copy, modify, merge, publish, distribute, sublicense, and sell copies. Under one condition of including the copyright notice and this permission notice in all copies or substantial portions of the software. This license grants me permission to freely fork the original OWASP PTK and use it as a foundation to develop OWASP PTK: Enhanced.
- Laws and regulation compliance: Development of penetration testing tools is legal in most countries. The development takes place in Hong Kong, which has no law or regulation that restricts the development and publication of penetration testing tools. Thus the development of OWASP PTK: Enhanced is completely legal. Some countries such as Germany and Australia have placed restrictions on the development and publication of penetration testing tools. (ICLG, no date) As of now, this project remains a private project. Further study on relevant laws needs to be made should this project be published. If necessary, regional lockout may be applied on countries that have placed restrictions on the development and publication of penetration testing tools.
- Data privacy and protection laws: The development of OWASP PTK: Enhanced did not include any additional data collection functionality. Respecting the goal of the original OWASP PTK of using the usage data collected only for the improvement of the software, the data collection functionalities of the original OWASP PTK were not removed. The original OWASP PTK collects only usage data from the users, it does not collect any data that may identify the users, such as personal information, IP address, user behavior, interests or preferences. As no data that may identify the users will be collected, this project is not affected by data privacy and protection laws.

- Liability: The original OWASP PTK has included this statement in their disclaimer statement, "The PTK team is not responsible for the outcome and results of the vulnerability scan". OWASP PTK: Enhanced too, adopts the same disclaimer statement. Penetration testers should be aware of their own liability in case any damage occurs during penetration testing.

7.2, Social

- Public trust: If penetration testing tools are used for malicious purposes, irresponsibility or unprofessionally, they may undermine the public trust in penetration testing tools and the field of cybersecurity by extension. As stated in the disclaimer statement, the OWASP PTK team and I are not responsible for the outcome and results of the usage of our tools. We trust the goodwill of penetration testers to use our tools responsibly, professionally and only with proper authorization. The existence of penetration testing tools is necessary for proactively identifying and patching potential security vulnerabilities before cyberattackers can exploit them, even at the risk of undermining the public trust.
- Impact on individuals and organizations: If penetration testing tools are used for malicious purposes, irresponsibility or unprofessionally, it can cause negative impacts on individuals or organizations whose systems were tested, such as unavailability of service, financial losses, reputational damage, or even job loss. As mentioned before, we trust the goodwill of penetration testers not to use our tools for malicious purposes. Documentation, tutorials, and user support will be provided to ensure the users understand the functionality of OWASP PTK: Enhanced and how to use it responsibly and professionally.

7.3, Ethical

- Security risks to the users: As browser extensions can augment or alter the functionality and behavior of the web browser, they may create new vulnerabilities to the web browser even if the browser extension was legitimate and developed with no malicious intent, especially extensions that require many permissions. OWASP PTK requires many permissions in order to function, such as access to cookies, local storage, scripting and request handling. If hijacked, it may cause negative impacts to the users, such as victim to data theft, homepage hijacking or in the worst case, as a medium for virus infection. Thus, OWASP PTK: Enhanced was developed without any additional permission or dependency on third party services. Designed to be as self contained as possible to minimize the risk of being hijacked.

- Accessibility: The OWASP PTK: Enhanced shares the same goal of improving the security of software worldwide as the OWASP. To achieve this, the tool should be made accessible to users of the widest range possible. Respecting the open source and free to use model of the original OWASP PTK, OWASP PTK: Enhanced too will be open source and free to use. Documentation, tutorials, and user support will be provided to ensure the users understand the functionality of OWASP PTK: Enhanced as well as widening their knowledge in the field of cybersecurity. While the original OWASP PTK: Enhanced only provides an English interface, more language options may be added in the future to enable easier usage for users not proficient in English.

7.4, Professional

- Competence: As a free software, I do not have the obligation to meet the expectation of the end users on the competence of OWASP PTK: Enhanced. However, incompetent or unauthoritative publication of software or documents may undermine the public trust in the field of cybersecurity. While I am not yet a qualified expert in the field of cybersecurity, I will do my best to develop OWASP PTK: Enhanced with the highest quality I can. Continuation of development and expansion of compatibility are planned, as they are to be expected by end users for modern day software. If possible, peer review with qualified experts in cybersecurity or integration with the original OWASP PTK may be conducted.

8, End-Project Review and reflection

After an evaluation on the project result and the project plan, I can conclude that the project was completed successfully at this stage. The objective of designing and developing a forked and improved version of the OWASP PTK has been achieved. 5 new major features are all successfully added, thoroughly tested and proven to be effective. With these newly introduced features, OWASP PTK: Enhanced can better stay on pair with the standalone software alternatives. However, this will likely not be the end of this project as continuation of development is planned for the future.

8.1, Best Aspects

In this project, I have challenged myself with both developing a functional and competence browser extension and functionalities of penetration testing tools. Before this project, I did not have abundant experience to take on either of the challenges, I had to learn on the fly as I worked on this project. At the end of this project stage, I am thrilled and motivated by the success of this project and the knowledge and skill I have acquired. I now have experience and a solid understanding in the development, security and application of browser extension and web application. As well as widened knowledge in the field of cybersecurity, particularly penetration testing. Yet, I still have a long way to go before I become a master in the field of cybersecurity. And even if so, the landscape of technology and security threats are ever changing. The study in cybersecurity has no end and one must never stop learning.

8.2, Worst Aspects

Due to the limitation of time for study and development, as well as my lack of experience and knowledge on the technology required for the completion of this project at this stage, some of the planned functions and features have been discarded through the development cycle of this stage. The details and reasons for most of the discarded functions and features are documented in the next section. However, I am not saddened by the lost opportunity of including those discarded functions and features in OWASP PTK: Enhanced, for they are now part of the plans for future development and improvement. I have no regret with the decisions I made with this project, for I have given it my best effort and decisions I made were as good as they can be with my limited experience and knowledge at the time.

8.3, Discarded Functions and Features

- Cross browser support: A feature that holds as much significance and importance as the successfully added features. It was planned for OWASP PTK:

Enhanced because cross browser compatibility and cross platform compatibility is one of the main advantages browser extensions have over standalone softwares. For now the reason for only supporting the Firefox browser was because Firefox browser is my favorite and preferred web browser. After the implementation of the interceptor of OWASP PTK: Enhanced, I realized how differently different browsers handle requests. The implementation of interceptor requires substantial and specific alteration to the way Firefox handles web requests. To make OWASP PTK: Enhanced compatible with other web browsers, the interceptor as well as other features needs to be modified specifically for each different browser. This will drastically increase the time requirement for implementation and testing for each new feature introduced in OWASP PTK: Enhanced. Yet, ensuring cross browser compatibility will be a priority in future development and improvement plans.

- Repeater: Featured in Burp Suite, repeater is an upscaled implementation of the Dirbuster. It encapsulates all the functionality of Dirbuster while being to support more advanced testing scenarios. It is also much more complex in implementation and customization. Due to the limitation of time and knowledge, I could not establish a plan of implementation for the repeater that can be finished within a reasonable amount of time. Instead, I compromised with the downscaled implementation of the repeater, the Dirbuster. Like cross browser compatibility, it will be a priority in future development and improvement plans.
- Wordlist importing function for the Dirbuster: This is a function that I have successfully implemented but later removed. When planning for the implementation of the Dirbuster, I have decided enabling the users to import their wordlist in TXT file directly to the interface will be one of the non-functional requirements for the Dirbuster. However, during the unit after successful implementation, I found out that using the file browser will cause the extension popup to lose focus and will close itself. The wordlist will not be loaded into the user interface as the user interface was closed. There is no workaround to this issue as it is an intended feature for Firefox and most other browsers. Although the import function works without any issue if the extension was instead accessed from a new browser tab or window. This feature was ultimately removed as it may cause confusion to the users who access the browser extension through the extension popup. As a remedy, a message that prompts the user to copy and paste their wordlist into the wordlist has been written as the placeholder text for the input box.
- Payload library: The R-Attacker feature of the original OWASP PTK enables penetration testers to simulate numerous kinds of attack against web applications, such as cross-site scripting, SQL injection, reverse shell and

server-side request forgery. In order to simulate the attacks, the user must craft the requests, which is time consuming. A payload library may enable penetration testers to speed up the process of request crafting by providing templates or payload presets, or even automate it. In theory, the payload library could be a viable addition to the original OWASP PTK that synergizes with its existing features. However, after a review on the project plan, I have decided this feature should not be implemented due to the risk of being misused and the potential damage it can cause.

There are more features and functions that were discarded during the project planning and scope definition process not listed above, most of them are discarded due to various reasons such as having low impact, reliance on third party services, high cost of maintenance or scalability issues. In the future, they may be reconsidered as my experience and knowledge expands and the time limitation is removed.

9, Conclusion and Future Works

In conclusion, the project has been a success in meeting the project objective and I'm satisfied with the results thus far. However, as mentioned in the previous section, there is still much work to be done to make the OWASP PTK: Enhanced reach the ideal state I envisioned. Particularly implementing cross browser and cross platform support, as it is one of the most prominent advantages of browser extensions. Once the development of OWASP PTK: Enhanced reaches its ideal state, I would like to send a pull request to the project leader of the original OWASP PTK, merging the two versions back to one. It has been a pleasure and honor for me to be able to make contributions to an open source project, a community and the world.

Reference

Shah, S. and Mehtre, B.M. (2014). *An overview of vulnerability assessment and penetration testing techniques*.

Alcorn, W., Frichot, C. and Orru, M. (2014). *The Browser Hacker's Handbook*. Available at:

https://books.google.com.hk/books?hl=en&lr=&id=IXr0AgAAQBAJ&oi=fnd&pg=PR15&dq=penetration+testing+browser+extension&ots=vgoDDYAsyx&sig=Fv_Pcpj0Ys2Huz6ZgLmT6_scef4&redir_esc=y#v=onepage&q&f=false

Mozilla (no date). *What are extensions?*. Available at:

https://developer.mozilla.org/en-US/docs/Mozilla/Add-ons/WebExtensions/What_are_WebExtensions

OWASP (no date). *OWASP Top 10*. Available at: <https://owasp.org/www-project-top-ten/>

OWASP (no date). *OWASP Penetration Testing Kit*. Available at:

<https://owasp.org/www-project-penetration-testing-kit/>

Wappalyzer (no date). *Wappalyzer*. Available at: <https://www.wappalyzer.com/>

Mozilla (no date). *HTTP cookies*. Available at:

<https://developer.mozilla.org/en-US/docs/Web/HTTP/Cookies>.

ENISA (no date). *Secure Software Engineering*. Available at:

<https://www.enisa.europa.eu/topics/critical-information-infrastructures-and-services/internet-infrastructure/secure-software-engineering>

OWASP (no date). *OWASP ZAP*. Available at: <https://www.zaproxy.org/>

PortSwigger (no date). *Burp Suite Scanner*. Available at: <https://portswigger.net/burp>.

PortSwigger (no date). *What is clickjacking?*. Available at:

<https://portswigger.net/web-security/clickjacking>

Robles, G. and Jes'us, M. (2012). *A Comprehensive Study of Software Forks: Dates, Reasons and Outcomes*. Available at:

<https://dl.ifip.org/db/conf/oss/oss2012/RoblesG12.pdf>

Atlassian (no date). *Agile Best Practices and Tutorials*. Available at:
<https://www.atlassian.com/agile>

Atlassian (no date). *What is version control?* Available at:
<https://www.atlassian.com/git/tutorials/what-is-version-control>.

Mozilla (no date). *MVC*. Available at:
<https://developer.mozilla.org/en-US/docs/Glossary/MVC>

Open Source Initiative (no date). *The MIT License*. Available at:
<https://opensource.org/license/mit>

ICLG (no date). *Cybersecurity Laws and Regulations*. Available at:
<https://iclg.com/practice-areas/cybersecurity-laws-and-regulations>

Mozilla (no date). *User-Agent*. Available at:
<https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/User-Agent>

Shar, L.K. and Tan, H.B.K. (2012). *Predicting common web application vulnerabilities from input validation and sanitization code patterns*. Available at:
<https://dl.acm.org/doi/abs/10.1145/2351676.2351733>

Google (no date). *Google Gruyere*. Available at: <https://google-gruyere.appspot.com/>

OWASP (no date). *OWASP Juice Shop*. Available at:
<https://owasp.org/www-project-juice-shop/>

Heroku (no date). *Heroku: Cloud Application Platform*. Available at:
<https://www.heroku.com/>

