APPLICATION SECURITY TOOLKIT FOR SMALL AND MEDIUM SIZED BUSINESSES

A DISSERTATION SUBMITTED TO MANCHESTER METROPOLITAN UNIVERSITY
FOR THE DEGREE OF MASTER OF SCIENCE
IN THE FACULTY OF SCIENCE AND ENGINEERING

2021

By
Luke Walker
Department of Computing and Mathematics

Contents

List of Tables

List of Figures

Abstract

Small medium-sized enterprises (SMEs) constantly struggle with demands from legislation and customers to ensure that their IT are running securely. For example, with the recent COVID-19 pandemic, many workers have moved to remote working causing the cyber attack surface to increase. The research conducted shows that SMEs, especially the smaller businesses of 1-10 employees, are especially vulnerable due to financial limitations and a lack of technical expertise. They will often have to outsource their IT provision and not have an internal IT department at all. In terms of the demands, impact of insecure IT in breach of various laws regarding information security, such as GDPR. The project looked at a range of publicly available application security programs and identified major weak points in the designs of the applications. They all expected the operator to be IT skilled and proficient. The purpose of this research is to investigate ways the information security industry can help improve cyber security provision for SMEs. The author of report wanted to explore the ways the industry could help to small business owners. An application security toolkit (AST) was built which would allow for the information security community to contribute, whilst making it easy for less-technically proficient people to reach out for support.The application was evaluated on ease of use and the level of technical understanding required to use the product. It was found, while programming issues were discovered in the product, it outputted data in a less technical way, making it easier for small business owners to get support. In conclusion, it is possible to create more appropriate application security tools aimed at SMEs, non technical members of staff require non technical outputs and if possible, the ability to contact a service desk to advise them on the steps to take to fix the issue reported.

Declaration

No part of this project has been submitted in support of an application for any other degree or qualification at this or any other institute of learning. Apart from those parts of the project containing citations to the work of others, this project is my own unaided work. This work has been carried out in accordance with the Manchester Metropolitan University research ethics procedures, and has received ethical approval number 34777.

Signed:

Date:

Acknowledgements

Stephen Doswell - Project Supervisor

Abbreviations

SME	Small and Medium
NCSC	National Cyber Security Center
POC	Proof of Concept
XSS	Cross Site Scripting
SQLi	SQL Injection
SSRF	Server Side Request Forgery

Chapter 1

Introduction

The idea behind this project is to preform research into currently available vulnerability scanners and how they function and then analyse what threats currently face UK based SMEs, especially software development companies. The research will then be analysed and a security toolkit application that's open source, community driven and on top of all, easy to use will be produced to help SME's face the extremely complex world of cyber.

1.0.1 Background

There is little support for small and medium sized businesses who wish to conduct trade online. With these type of companies having an extremely small information technologies department, or non at all, it's difficult for these companies to ensure they're staying safe and secure online. Cyber Security can become extremely expensive. A well known Cyber Security company, known as Bulletproof charge between £1000 - £20000 depending on the scope, which companies will struggle to afford *Penetration Testing Services – Certified Pen Testers UK* (n.d.).

Taking the above into account, there needs to be a way that small/medium sized businesses can protect themselves and stay in line with the NCSC guidelines regarding Cyber Security. In such case this dissertation will look into currently available software packages and analyse the cost, ease of use and the amount of false positives it picks up. Once those topics have had data collected about them, A toolkit shall be written which can automatically test for vulnerabilities, including but not limited to:

- OWASP Top 10 for 2021
- Binary exploitation (Buffer Overflows, ret2libc)

- Anti Malware detection capabilities

The toolkit, by design, should be easy to use, even for a programmer with limited security knowledge. The application should output any vulnerabilities, if detected within the application/website and have an easy to understand methodology on how to fix/mitigate the issue that has been discovered. It is well known that quite a lot of the well known vulnerability scanners (Qualys, Acuneticx) can be difficult to interpert without training given before hand, this shouldn't be the way forward as SME's with little or no security minded staff won't understand what is needed to make their business secure, which then leads them to ignoring potentially serious issues or paying large amounts of money for a contractor. In most cases, businesses with sub £100,000 net profits are not going to spend thousands of pounds to hire a professional to fix issues for them, which can lead to customers information being breached.

1.0.2 Aims and Objectives

This project will investigate the currently available vulnerability scanners and code analysis tools and attempt to develop a community driven, open source replacement which is easy to use for individuals and SMEs. To achieve this aim, a number of objectives have been set. These are as follows:

- Terms of Reference – Provides an overview of the project
- Develop a project plan to be followed at each stage in the lifecycle of the research and development
- Analysis of current tools – Gather information on already known applications and preform critical analysis on them
- Analysis of current threats facing SMEs – Gather information on threats that face SMEs and preform critical analysis, identifying worries these companies have
- Undertake research into the history of Application security toolkits and thereby undertake a review of currently available products
- Draw up formal requirements for the AST
- Design the AST based on the identified requirements of SMEs.

- Develop the application – Script the application and provide it to a select SME for user review
- Final Report – Document the stages of the project

In order to determine if this tool will be effective, a hypothesis was set: “Cyber security issues within SMEs stem from the fact they don’t have access to proper security testing facilities, rather due to technological knowledge required, or financial impact these tools cost”

Chapter 2

Literature Review

2.0.1 SME Importance to the UK economy

There were 6.0 million SMEs in the UK in 2020, accounting for more than 99% of all businesses. The micro business has 0-9 employees. In 2020, there were 7.7 million micro-businesses in the UK, making up 5% of all businesses. FSB (2020) Although most businesses in the UK employ less than 10 people, this type of business has only 33% employment and 21% business. There are 8,000 large industries (A business with more than 205 employees), of which the industry has a 0.1% share but 39% employment and 48% turnover. It is estimated that SME's account for three fifths of the employment within the UK and these numbers are rising even with the COVID-19 pandemic devastating the global economy. This report is extremely useful information when trying to choose which organisations to promote the application security toolkit too, for example the fastest growing sector in 2020 was the transport and storage sector. This is more likely due to most people not shopping within stores during the Covid 19 pandemic and defaulting to buying products they need online through vendors such as Amazon. This information tells us that we might need to target the application to slower computers, since transport and storage companies won't have powerful workstations like a software development company might. However a single person enterprise was the most popular type of business within the UK during 2020. Single person enterprises account for 76.3% of all the businesses within the UK economy and as such won't have a team of IT Security professionals on hand to validate their website/Apps security. This is not good at all, such companies will potentially skip corners when it comes to designing their website which can introduce attack vectors for malicious entities, if the company wants a custom designed website, there is no guarantee that

the application will not be filled with bugs. Most Businesses of this size however do rather not have an online presence at all or will use easily drag and drop site builders from companies such as squarespace or GoDaddy. This is useful in telling us that that we don't need to target these sites too much as they've been programmed by extremely experienced people. On the other hand bugs can still exist in any computer system, but on the occasion a bug is found within a website construction company it is not the clients fault and as such those types of bugs should be directly submitted to the company managing the website. This rise is attributed to self employment skyrocketing during the COVID-19 pandemic and with the lack of job opportunities, people have been driven to making their own business, with 4.5 million businesses having 0 employees. As mentioned before, it would be recommended that a self employed person paid a well known company to develop and manage their website for them, as this will reduce the chances of vulnerabilities being present in the code.

2.0.2 Challenges facing SME's

The most important concerns for SMEs within 2020 were the new Coronavirus regulations and the new Brexit regulations. In fact, 51% of small business owners say the current political situation makes it difficult for them to pursue their plans to grow their business. The pandemic has spurred great change for businesses, such as digitization, e-commerce and telecommuting, but the shift will be permanent and it is unclear how much they will affect different parts of the economy. Compared to the COVID-19 pandemic, the impact of changes in the UK's trade relations with the EU is less pronounced, with far fewer people and businesses having a direct impact. In the short term, importers and exporters are used to doing business outside of the EU Customs Union and face trade disruptions, which can directly effect the annual turnover for SME's. This point could potentially effect the amount of money companies have to spend on Cyber Security training, leaving themselves wide open to threats. These changes do not impact the large players such as Amazon, as they don't have any stores and have vast quantities of stock already available within the UK. Large businesses can also afford the new import costs associated with certain items, such as Alcohol.

2.0.3 Overview of Cyber challenges facing the UK

This is a large part of the economy who struggle with keeping their internet facing applications, such as websites or mobile apps secure from attackers. The UK NCSC

has provided information for SME's to protect their businesses online capabilities in the currently growing security threat of Ransomware. Additionally, another common method affecting all UK business and government is phishing attacks, phishing attacks have been documented by some APT's as the initial method for entering a computer network, along with that as we'll discuss below, It's an extremely financially rewarding method of making money and only a small amount of IT knowledge is needed to pull off a phishing attack. This makes it extremely lucrative for Cyber Criminals, due to the anonimity and easy deployment, anyone, can be targeted. Sadly for businesses and people, this attack mainly targets the person, not the computer directly via a method called social engineering, this makes it harder for IT teams, big or small to protect the business from attacks, even with sophisticated software packages.

2.0.4 Key Cyber issues facing UK SME's

Businesses from the UK are facing new threats that didn't exist 10 years ago, as we all move to an online world SME's are struggling to keep up. A recent survey done by the Department for Digital, Culture, Media and sport reveals that 43% of SME's experienced a cyber attack within the last 12 months, with 19% of charities also facing cybersecurity attacks. Johns (n.d.). The report also reveals that 98% of all UK businesses have an online presence, whether that be a web shop, website, etc. The main threat that all businesses in the UK face currently is CryptoLocker and Ransomware attacks, most SME's will struggle to recover data from a successful ransomware attack due to the cost and technical knowledge needed to combat these style of attacks. The Ransomware threat is steadily growing and threat actors are not attacking small businesses any more and instead are opting for supply chain attacks against larger managed service providers, which directly cause business shutdowns for SME's. The second biggest threat currently lies within Phishing emails, attackers are increasingly getting better at creating fake emails to target smaller businesses, with most being whaling attacks. A whaling attack directly targets the CEO of a company for financial gain for the attacker, Extortion is also on the rise among SME's which generally work similarly to Phishing, apart from the attacker may attempt to Dox or find personal information about an employee of a SME and threaten to leak it. The most common form of this type of attack is sextortion, where an attacker will claim to have private images of a person and demand a ransom for them to be deleted. The most common forms of attacks as just discussed are relatively easy to pull off with underground and darknet

forums allowing for the trade of exploit kits. These kits make it relatively simple to deploy a phishing campaign against a given target. there are many types of phishing kits available ranging from Office 365, Facebook, Twitter, Instagram and more generally at quite a low cost. these kits also come with a full walkthrough on how to deploy the kit to target a certain organisation. Due to the relatively low skill needed to deploy these kits, anyone with intermediate computer and server knowledge Is able to buy and run a phishing campaign, or ransomware. The kits are usually sold for relatively cheap, as discussed in a report by bleeping computer a phishing kit can be picked up for around $25 per kit. Ilascu (2020). The National Cyber Security Centre *The Cyber Threat to UK Business* (n.d.) has published information regarding the threats that companies should be looking out for, one note worthy example of this, is that threat actors are learning and sharing information with each other, which is a relatively new practise. It's interesting to see within this report that the most common types of cyber crime isn't happening with 0day exploits, it's mainly low level, easy to pull off attacks. The main attacks listed in the report are distributed denial of service attacks, phishing and financial trojans. The NCSC recommends companies to ensure they have anti malware software on their devices, Data is securely backed up to more than one place and secure passwords are used. *Small Business Guide: Cyber Security* (n.d.)

2.0.5 Overview of commercially available Vulnerability scanners

Burp Suite Professional

Burp Suite professional is a web proxy and a vulnerability scanner that allows the user to import a custom CA certificate into their web browser which allows the application to proxy requests and responses that the server and web browser make. The application also allows for requests to be sent to the vulnerability scanner for an active or passive scan. Other features of the applications include;

- Intruder
- Repeater
- Sequencer
- Decoder
- Comparer

- Collaborator

From the top, the Intruder is effectively a brute force module, it can be used for password testing along with payload spraying. The application asks the user to use pre-programmed in wordlists, or they can add their own. Once the user hits start, the application will spray requests to the web endpoint to attempt to find a vulnerability.

The repeater functionality allows a user to send a request across from the proxy and then repeat the request, it allows the user to fine tune payloads and to test for web application firewalls, amongst other things.

The Sequencer allows you to load in a list of tokens, for example JWT tokens and sprays them at the target. This is useful for testing vulnerabilities that may lie in the authentication application on a website.

The Decoder allows the user to paste in a string that might be encoded in certain formats, such as Base64 and URL/HTML and attempts to decode it back to a readable format. This can be good for finding secrets stored insecurely in Base64

The Comparer allows you to compare 2 different web requests. It is automatic and can do word or byte comparisons. Useful for testing if certain payloads give a different response from the website in testing.

The Collaborator allows for the generation of web URLs, when the web URL is accessed, the Burp application will display what type of request was received. The collaborator allows for all HTTP methods, along with DNS logging. It allows a user to test issues such as Server side request forgery without having to run their own web service.

There is also a BApps marketplace, where developers can submit their own modules for customers to download, there seems to be modules for nearly about everything which add additional functionality to Burp Suite.

When the application discovers a potential vulnerability, it is reported to the user by displaying the payload in the context of the website. The application also displays an Issue background which explains to the user what the vulnerability means in the context of the website along with an Issue remediation section, which attempts to inform the user how the vulnerability can be fixed on the server.

The application has an easy to use GUI and is coded in Java. It additionally allows the user to save the project so they can continue working on it at a later date.

I was unable to complete a scan using Burp Suite scanner, while PortSwigger issued me a 30 day trial to the software, their web application was unable to send me a password to authenticate, therefore Burp Suite ended with a did not finish, sadly.

Nikto/Wikto

An extremely old vulnerability scanner, It can be used from the command line or via a web browser. Released in 2001 with the stable release as of writing being 2.1.6, released on the 9th of July, 2015. The source code for Nikto is open source, however the data files it relies on to preform vulnerability analysis are not. The application runs various checks for known vulnerabilities such as CVE's, alongside checking for any CGI directories and files.

Within testing against DVWA, Nikto found 40 issues and submitted 8724 requests against the remote DVWA server. Nikto was able to detect the server was vulnerable to CVE-2014-6271, or shellshock, however apart from this, Nikto only discovered potential issues. Full terminal output is below

```
- Nikto v2.1.6
---------------------------------------------------------------------
+ Target IP:          192.168.102.131
+ Target Hostname:    192.168.102.131
+ Target Port:        80
+ Start Time:         2021-10-22 01:30:55 (GMT-4)
---------------------------------------------------------------------
+ Server: Apache/2.2.14 (Unix) DAV/2 mod_ssl/2.2.14 OpenSSL/0.9.81
+ Retrieved x-powered-by header: PHP/5.3.1
+ The anti-clickjacking X-Frame-Options header is not present.
+ The X-XSS-Protection header is not defined. This header can hint
+ The X-Content-Type-Options header is not set. This could allow th
+ Cookie PHPSESSID created without the httponly flag
+ Cookie security created without the httponly flag
+ Root page / redirects to: login.php
+ Server may leak inodes via ETags, header found with file /robots.
+ mod_apreq2-20090110/2.7.1 appears to be outdated (current is at l
+ OpenSSL/0.9.8l appears to be outdated (current is at least 1.1.1)
+ Apache/2.2.14 appears to be outdated (current is at least Apache/
+ Perl/v5.10.1 appears to be outdated (current is at least v5.20.0)
+ mod_ssl/2.2.14 appears to be outdated (current is at least 2.8.31
+ mod_perl/2.0.4 appears to be outdated (current is at least 2.0.8)
+ PHP/5.3.1 appears to be outdated (current is at least 7.2.12). PH
```

```
+ Apache mod_negotiation is enabled with MultiViews, which allows a
+ OSVDB-877: HTTP TRACE method is active, suggesting the host is vu
+ mod_ssl/2.2.14 OpenSSL/0.9.8l PHP/5.3.1 mod_apreq2-20090110/2.7.1
+ OSVDB-112004: /cgi-bin/printenv: Site appears vulnerable to the '
+ OSVDB-112004: /cgi-bin/printenv: Site appears vulnerable to the '
+ OSVDB-3268: /config/: Directory indexing found.
+ /config/: Configuration information may be available remotely.
+ OSVDB-12184: /?=PHPB8B5F2A0-3C92-11d3-A3A9-4C7B08C10000: PHP reve
+ OSVDB-12184: /?=PHPE9568F36-D428-11d2-A769-00AA001ACF42: PHP reve
+ OSVDB-12184: /?=PHPE9568F34-D428-11d2-A769-00AA001ACF42: PHP reve
+ OSVDB-12184: /?=PHPE9568F35-D428-11d2-A769-00AA001ACF42: PHP reve
+ OSVDB-561: /server-status: This reveals Apache information. Comme
+ OSVDB-3092: /phpmyadmin/changelog.php: phpMyAdmin is for managing
+ OSVDB-3092: /phpmyadmin/ChangeLog: phpMyAdmin is for managing MyS
+ OSVDB-3233: /cgi-bin/printenv: Apache 2.0 default script is execu
+ OSVDB-3233: /cgi-bin/test-cgi: Apache 2.0 default script is execu
+ OSVDB-3268: /icons/: Directory indexing found.
+ OSVDB-3268: /docs/: Directory indexing found.
+ OSVDB-3092: /CHANGELOG.txt: A changelog was found.
+ OSVDB-3233: /icons/README: Apache default file found.
+ /login.php: Admin login page/section found.
+ /phpmyadmin/: phpMyAdmin directory found
+ OSVDB-3092: /.svn/entries: Subversion Entries file may contain di
+ OSVDB-3092: /phpmyadmin/Documentation.html: phpMyAdmin is for man
+ /CHANGELOG.txt: Version number implies that there is a SQL Inject
+ OSVDB-3092: /phpmyadmin/README: phpMyAdmin is for managing MySQL
+ 8724 requests: 0 error(s) and 40 item(s) reported on remote host
+ End Time:           2021-10-22 01:31:18 (GMT-4) (23 seconds)
---------------------------------------------------------------------
+ 1 host(s) tested
```

OWASP ZAP

OWASP ZAP is extremely similar to Burp Suite, the application is coded in Java and can act as a web proxy to capture requests. As with Burp these requests can also be edited in transit. The application also features a vulnerability scanner, split down into

different modules.

The first module is an active scan, which just fires payloads at a given target to see if any of them cause the web application to interact in a strange manner, the application by nature is extremely loud and will generate lots of web traffic, unlike Burp suite, the application isn't "smart" in the way it sends payloads. For a given vulnerability type, the application might send 3000 requests and get nothing back. There is also a passive scanner, which by definition, will run several scripts to determine things such as the web server technology (Apache, Nginx), DNS records and other general informational tasks. Alongside these there is a built in spider, much like Burp along with a AJAX web crawler.

Alongside this, there is also a marketplace where users of the application can extend ZAP with community developed plugins.

A special feature that ZAP has is the ability to interact with it in daemon mode via a REST API, a feature which is rarely seen in commercial vulnerability scanners. The API allows for users to make web requests to the ZAP daemon and in turn trigger events such as running active scans.

ZAP, much like Nikto, is open source. Running ZAP against the DWVA instance, it was able to discover 8 alerts, with most of them being non exploitable issues. ZAP failed to detect CVE's that Nikto had no problem doing.

2.0.6 Critical review of Vulnerability Scanners

Many vulnerability scanners generally operate under the guise of "you get what you pay for", however it's very dependant on the type of vulnerabilities you want to scan for, for example, complex and application based vulnerabilities usually require expensive vulnerability scanners such as Acunetix and Qualys, as these are designed to pick up complicated vulnerabilities and additionally analyse the application. In the terms of other requirements, there are many free and accurate CVE scanners, such as Nuclei, which uses a template based system for detecting basic exploits such as XSS along with known CVE's with public proof of concepts.

In terms of paid vulnerability scanners however, with Qualys, the application requires you to run it on the "Qualys Cloud" with a internal server setup within your organisation to assist the cloud instance with certain scanning. This is obviously way to complicated and expensive for SME's to attempt to deploy as a subscription to Qualys is needed, along with a spare internal server that has access to the internal LAN. Other vulnerability scanners can return complicated results that can be hard to understand

unless you've got a education in web development or information security.

In terms of the 3 reviewed above, Nikto was able to pick up vulnerabilities more accurately that ZAP was able to, to demonstrate this, Walker et al crafted an array of virtual machines, placed on their own vLAN and ran each application against DVWA *Damn Vulnerable Web Application* (n.d.).

2.0.7 Requirements for the AST

The below points are graded from High to Low based on application needs.

- Simple to use and easy to understand the results to be supported by potential user feedback - Medium
- Implement an API for community developed modules - Low
- Implement a robust product (commercially viable) that has a maximum of 3 medium level bugs - Medium
- Ensure the product has an acceptable level of performance (Depending on the size of the scan) - Medium
- Implement example accessibility features eg: night mode/font size - Low
- Implement graphical user interface to support usability - High
- Implement widely recognised industry leading web testing methodologies, eg: OWASP for Web - High
- Implement widely recognised industry leading C app testing methodologies, eg: CVE for application support - High
- Cover basic C family application testing (Buffer Overflows, String formats) - High
- implement output/report system, eg:(Export in Github/Bugzilla format) - Medium

Chapter 3

Development

The application, as already mentioned in this paper will be coded using Python3.7 and Qt Creator/Designer for the GUI. The VirusTotal API v3 will be implemented into the application to provide malware analysis, I could have chosen to create my own database of known malware hashes, however this would be a weakness of the application, it's much better to use VirusTotal who have a database of over 2 billion file hashes.

The application will use a module type interface, where developers can easily integrate new features into the application. To do this I needed an algorithm that would automatically import all the module files from a given directory. This was achieved with the following algorithm:

```
import pkgutil

__path__ = pkgutil.extend_path(__path__, __name__)
for importer, modname, ispkg in pkgutil.walk_packages(path=__path__,
                                    prefix=__name__+'.'):
    __import__(modname)
```

In the end, this algorithm caused multiple problems with importing the modules. The main issue being that I couldn't get the Python application to see the modules in the folder. I switched to using a different method. I created a init python file within the modules folder with the following code

```
    __all__ = ["gui", "request"]
```

This method isn't as clean as the other one, as it would require the community to firstly add their module into the modules folder, then update the above array with the name of their script. However the import is done automatically within the main.py file.

This could potentially invoke a security vulnerability within the application, since it will automatically trust any script within the modules folder, however at this stage this isn't something Walker et al needed to consider in the development lifecycle.

In terms of code optimisation, the code/application at the moment is single threaded. This means that the application will use a single CPU thread to execute each function. In comparison to other tools, multi threading is enabled, which allows the "scanner" thread to run on a different CPU core and as such allows for the application to cross talk between CPU threads. In this statue, tested applications will allow for graphical user interfaces that automatically update when a vulnerability is found. In the current Python development branch of the application, the application currently runs in single threaded mode, which means users might face application lock ups during scanning sessions, especially when live or active scans are being preformed. This was a issue I spotted when originally implementing the software scanner side of the application. Since this side of the application uses a API, more specifically VirusTotal to process requests, the application locks up for a couple seconds while the API returns the data to the machine. This is an issue I'd aim to fix for a full release version of the software, but as mentioned below, I used rapid prototyping to deploy/write the application.

In terms of the software scanning section of the application, as mentioned above it uses the VirusTotal API to manage the detection of malware and potential threats. The application allows a user to select a file from their system, it then computes the SHA256 hash of the file and sends that to the API, instead of trying to upload massive files to the VT API.

The VT API then responds with the data it has on the hash in question, this data is returned in JSON format, then parsed by the toolkit using the JSON python library, the application then takes this data and crafts a pie chart with it using the PySide GUI designer that I've written the rest of the application in. This will be easier to read for the SME clients I'm targeting this application towards.

An issue that might crop up later during deployment of the application is that the Vt API requires and API key to function, during testing I've used my own API key which is limited to the amount of requests that the key can make to the VT servers. Since this application has been designed for SME's there needs to be an easy way to get an API key into the application, this is an issue Walker et al would need to think about if this application was to be sold as a commercial product.

The vulnerability scanner currently detects cross site scripting payloads within a web response, this process is done via a detection algorithm, which currently uses a

complicated regex to pass the data that is returned from the web server. To test the scanner, Walker et al wrote a vulnerable PHP script with two reflections, upon running the vulnerability scanner against my test environment the script discovered the two reflections and alerted me accordingly, due to the rapid prototyping design of this project other vulnerability types are not currently being detected however adding support would be relatively simple as it would just require adjusting the regex appropriately. This is not the way most commercial vulnerability scanners work as most of them feature a built in chromium based sandbox which can detect for arbitrary JavaScript execution, which is something Walker et al could consider developing into the application outside of the rapid prototyping development I've used to write this program.

Chapter 4

Evaluation

4.0.1 Evaluation of the product

When undertaking this project, The author had a couple of key ideas in mind. The author agrees these ideas have been met. However, performance wise the author would have optimized the way he approached the application programming. Due to the lack of public information on the libraries used, the author found it difficult to implement some of the features that had previously been discussed, which the author has gone into detail about below.

In terms of detection rates and functionality, the author deployed a PHP interpreter to serve a PHP file that was vulnerable to cross site scripting as part of the testing stage. The PHP script included 2 reflected cross site scripting issues, which would reflect into the document object model, causing arbitrary javascript execution on the client. Both of the vulnerabilities got picked up by the application, so in terms of detection methodologies, the author is proud to publish the work done here. The application also returned the data in a simple and easy to read format with contact information for a security analyst to help them fix the issue in their computer system.

The application scanner (binary) worked just as planned. While the author wasn't able to implement the fuzzer on time for the submission of this research, the malware detection was drastically improved up on. The malware detection currently uses around 50 anti malware detection platforms and compiles the results together, giving the operator a quick and easy to understand pie chart of the detection status of the file provided to the application. This can allow the operator to make an informed choice on if they need to deploy a third party anti malware solution, eg: MalwareBytes on the affected machine.

The application in this sense is harder to manipulate by malware as the application would require a rewrite in order for false results to be given back to the user. Since the application has been programmed in Python, it'd be relatively simple to malware to inject malicious code into the application, this isn't a vulnerability on my behalf per say, but more of a technical limitation of the Python programming language. This vulnerability would more than likely be present in most languages as long as the malware had control over the memory regions the application was running in, so the author considers it a none issue.

The application was developed using rapid prototyping, which allowed the author to focus completely on one area of the application before moving onto developing another section. This, in the author's opinion, was the best way to approach the programming section of this research paper, mainly due to the application not being a complex piece of code. The author considers the application to be functional, and with the aims set, the application could easily be developed into a full commercial product. This is down to how the author programmed the scripts, with little efforts being needed to add in additional features and detection of other vulnerabilities being simple to add for any intermediate programmer, the author thinks he's passed his target of developing a community based scanning application.

The author thinks in regards to programming the application, the author overestimated how many features he would be able to implement in the time period the author had, this wasn't a lack of knowledge around the Python language per say, but more due to the fact the author was using massive libraries such as PySide2 he had never used before, the author however thinks he picked up the package rather well, especially due to the small amount of examples and code available online for the library. Most of the code and examples the author found had been written in C++/C, so a couple parts of the code, especially the pie chart used the author's limited C++ programming skills to port the code into Python, something which the author has not ever really touched on.

4.0.2 Evaluation of the Project

The project is considered by the author to have gone relatively well, personal issues aside, the author was able to deploy the application via the rapid prototyping method discussed in the previous subsection, along with touching up on key issues within the research that other academic works haven't discussed as of yet. The author believes the research conducted as part of the Masters degree will be more beneficial to industry than the application, this is due to the application currently being in a proof of concept

state, rather than a finished application that could be deployed to customers/SME's as a commercial product. Within the author own personal habits, he would have preferred to get the programming section of the project completed before the write up section, this is due to the project manager recommending the author to continue with the writing of this dissertation. While the author agrees this was a good choice on the project manager, a choice in which saved me a lot of time, the author frequently didn't supervene to it, due to personal issues.

There was some issues with PortSwigger giving me a license to Burp Suite Enterprise so I could test the scanning methodologies within the application, this of course causes issues within the literature review section of the application, in any case, the author has completed the research to the best of his abilities, but it was worth mentioning this.

In regards to time management, apart from the reasons mentioned above, the author generally kept to my own time management criteria, in which the author had set himself goals at the beginning of every week to complete certain areas of the report and application, which would later be confirmed by my project manager on Thursdays. The application development did lag behind every so often, mainly due to features taking longer to implement, which didn't cause issues per say, but in my next report/project the author is going to make sure the author gives myself extra buffer time to make sure that features can be implemented to schedule.

In terms of the workshops at University, the author found the workshop with Jim O'Shea related to the word choices we make when writing to be the most useful one out of all of them, it helped inspire me to use different words to explain concepts and write more effectively, The author didn't find many of the workshops that helpful, since quite a lot of them seemed to recap concepts that'd already looked into for my BSc degree just over a year ago as of writing. The author personally thinks the wording of my report has improved due to the workshop that Jim hosted, the part that helped the most was splitting off into smaller groups and having a hands on debate with other University Postgraduate students about the topic we got assigned.

References

Damn Vulnerable Web Application (n.d.). URL: https://dvwa.co.uk/.

FSB (2020). *UK Small Business Statistics*. URL: https://www.fsb.org.uk/uk-small-business-statistics.html (visited on 06/07/2021).

Ilascu, Ionut (July 2020). *Over 1,300 phishing kits for sale on hacker forum*. URL: https://www.bleepingcomputer.com/news/security/over-1-300-phishing-kits-for-sale-on-hacker-forum/.

Johns, Emma (n.d.). *Cyber Security Breaches Survey 2021*. URL: https://assets.publishing.service.gov.uk/government/uploads/system/uploads/attachment_data/file/972399/Cyber_Security_Breaches_Survey_2021_Statistical_Release.pdf.

Penetration Testing Services – Certified Pen Testers UK (n.d.). URL: https://www.bulletproof.co.uk/penetration-testing.

Small Business Guide: Cyber Security (n.d.). URL: https://www.ncsc.gov.uk/collection/small-business-guide.

The Cyber Threat to UK Business (n.d.). URL: https://www.ncsc.gov.uk/files/The%20Cyber%20Threat%20to%20UK%20Business%20(b).pdf.

Appendix A

Source Code

The code below makes up the Security testing toolkit, developed as part of this project. Code is seperated into files and will not execute correctly without the structure stored on `https://github.com/0x0luke/Warlord`

Main.py

```
    # Loader

    from modules import *

    gui.setupUI()
```

GUI.py

```
    from PySide2 import QtCore, QtWidgets, QtGui
from PySide2.QtCharts import QtCharts
import sys
import os.path
from virustotal_python import Virustotal
from pprint import pprint
import hashlib
import json
import re
import requests

class WebAppWindow(QtWidgets.QWidget):
    def __init__(self):
        super().__init__()
        self.setWindowTitle("Web App Testing")
        self.resize(640,480)
        self.layout = QtWidgets.QVBoxLayout()
```

```
        self.UrlBox = QtWidgets.QTextEdit(self)
        self.UrlBox.setGeometry(QtCore.QRect(270, 90, 30, 31))
        self.UrlBox.setObjectName("UrlBox")
        self.UrlBox.setText("Place the URL here!")

        # file selector code
        self.filebtn = QtWidgets.QPushButton("Select a payload file
                                             to test")
        self.filebtn.setGeometry(QtCore.QRect(30, 90, 80, 23))
        self.filebtn.clicked.connect(self.getExploits)
        self.filePathBox = QtWidgets.QTextEdit(self)
        self.filePathBox.setGeometry(QtCore.QRect(10, 90, 30, 31))
        self.filePathBox.setObjectName("filePathBox")
        self.layout.addWidget(self.filebtn)

        self.buttonBox = QtWidgets.QDialogButtonBox(self)
        self.buttonBox.setGeometry(QtCore.QRect(10, 280, 621, 32))
        self.buttonBox.setOrientation(QtCore.Qt.Horizontal)
        self.buttonBox.setStandardButtons(QtWidgets.QDialogButtonBox
                                             .Ok)
        self.buttonBox.setObjectName("buttonBox")

        self.buttonBox.accepted.connect(self.accept)

        self.layout.addWidget(self.filebtn)
        self.layout.addWidget(self.filePathBox)
        self.layout.addWidget(self.buttonBox)
        self.layout.addWidget(self.UrlBox)

        self.errMsg = QtWidgets.QMessageBox(self)
        self.setLayout(self.layout)

    def accept(self):
        WebAppWindow.webRequest(self)

    def getExploits(self):
        fname = QtWidgets.QFileDialog.getOpenFileName(self, "Open
                                             File",
        "c:\\")[0]
        self.filePathBox.setText(fname)
        return fname
```

```
def webRequest(self):
    filename = self.filePathBox.toPlainText()
    url = self.UrlBox.toPlainText()
    with open(filename, encoding="utf-8") as file:
        payloads = file.readlines()
        for line in payloads:
            request = requests.get(url + line.strip())
            if request.status_code != 200:
                self.errMsg.setWindowTitle("Error!")
                self.errMsg.setText("The website gave us
                                                response code
                                                " + str(
                                                request.
                                                status_code) +
                                                 "!\nScanning
                                                won't work
                                                correctly!")
                self.errMsg.setIcon(QtWidgets.QMessageBox.
                                                Critical)
                box = self.errMsg.exec_()

            #Vuln detection logic
            XSSRegex = len(re.findall("<[^\w<>]*(?:[^<>\"'\s]*:)
                                            ?[^\w<>]*(?:\W*s\W
                                            *c\W*r\W*i\W*p\W*t
                                            |\W*f\W*o\W*r\W*m|
                                            \W*s\W*t\W*y\W*l\W
                                            *e|\W*s\W*v\W*g|\W
                                            *m\W*a\W*r\W*q\W*u
                                            \W*e\W*e|
            (?:\W*l\W*i\W*n\W*k|\W*o\W*b\W*j\W*e\W*c\W*t|\W*e\W*
                                            m\W*b\W*e\W*d|\W*a
                                            \W*p\W*p\W*l\W*e\W
                                            *t|\W*p\W*a\W*r\W*
                                            a\W*m|\W*i?\W*f\W*
                                            r\W*a\W*m\W*e|\W*b
                                            \W*a\W*s\W*e|\W*b\
                                            W*o\W*d\
```

```
W*y|\W*m\W*e\W*t\W*a|\W*i\W*m\W*a?\W*g\W*e?|\W*v\W*i
                                    \W*d\W*e\W*o|\W*a\
                                    W*u\W*d\W*i\W*o|\W
                                    *b\W*i\W*n\W*d\W*i
                                    \W*n\W*g\W*s|\W*s\
                                    W*e\W*t|\W*i\W*s\W
                                    *i\W*n\W*d\W*e\W*x
                                    |\W*a\W*
n\W*i\W*m\W*a\W*t\W*e)[^>\w])|(?:<\w[\s\S]*[\s\0\/]|
                                    ['\"])(?:
                                    formaction|style|
                                    background|src|
                                    lowsrc|ping|on(?:d
                                    (?:e(?:vice(?:(?:
                                    orienta|mo)tion|
                                    proximity|found|
                                    light)|livery(?:
                                    succe
ss|error)|activate)|r(?:ag(?:e(?:n(?:ter|d)|xit)|(?:
                                    gestur|leav)e|
                                    start|drop|over)?|
                                    op)|i(?:s(?:c(?:
                                    hargingtimechange|
                                    onnect(?:ing|ed))|
                                    abled)|aling)|ata
                                    (?:setc(?:omplete|
                                    hanged)|(?:avail
abl|chang)e|error)|urationchange|ownloading|blclick)
                                    |Moz(?:M(?:
                                    agnifyGesture(?:
                                    Update|Start)?|
                                    ouse(?:PixelScroll
                                    |Hittest))|S(?:
                                    wipeGesture(?:
                                    Update|Start|End)?
                                    |
                                    crolledAreaChanged
                                    )|(?:(?:Press)?
                                    TapG
```

```
estur|BeforeResiz)e|EdgeUI(?:C(?:omplet|ancel)|Start
                                    )ed|RotateGesture
                                    (?:Update|Start)?|
                                    A(?:udioAvailable|
                                    fterPaint))|c(?:o
                                    (?:m(?:p(?:osition
                                    (?:update|start|
                                    end)|lete)|mand(?:
                                    update)?)|n(?:t(?:
                                    rolselect|extmenu)
                                    |nect(?:ing
|ed))|py)|a(?:(?:llschang|ch)ed|nplay(?:through)?|
                                    rdstatechange)|h
                                    (?:(?:arging(?:
                                    time)?ch)?ange|
                                    ecking)|(?:fstate|
                                    ell)change|u(?:
                                    echange|t)|l(?:ick
                                    |ose))|m(?:o(?:z
                                    (?:pointerlock(?:
                                    change|error)|(?:
                                    orientation|time)
                                    change|fullscreen(
   ?:change|error)|network(?:down|up)load)|use
                                        (?:(?:lea|mo)
                                        ve|o(?:ver|ut)
                                        |enter|wheel|
                                        down|up)|ve(?:
                                        start|end)?)|
                                        essage|ark)|s
                                        (?:t(?:a(?:t
                                        (?:uschanged|
                                        echange)|lled|
                                        rt)|k(?:
                                        sessione|comma
                                        )nd|op)|e(?:ek
                                        (?:complete|
                                        ing|e
```

```
d)|(?:lec(?:tstar)?)?t|n(?:ding|t))|u(?:ccess|
                                      spend|bmit)|
                                      peech(?:start|
                                      end)|ound(?:
                                      start|end)|
                                      croll|how)|b
                                      (?:e(?:for(?:e
                                      (?:(?:
                                      scriptexecu|
                                      activa)te|u(?:
                                      nload|pdate)|p
                                      (?:aste|rint)|
                                      c(?:opy|ut)|
                                      editfocus)|
                                      deactivate)|gi
n(?:Event)?)|oun(?:dary|ce)|l(?:ocked|ur)|
                                      roadcast|usy)|
                                      a(?:n(?:
                                      imation(?:
                                      iteration|
                                      start|end)|
                                      tennastatechange
                                      )|fter(?:(?:
                                      scriptexecu|
                                      upda)te|print)
                                      |udio(?:
                                      process|start|
                                      end)|d(?:
                                      apteradded|
                                      dtrack)|
                                      ctivate|
                                      lerting|bort)|
                                      DOM(
```

```
?:Node(?:Inserted(?:IntoDocument)?|Removed
(?:
FromDocument
)?)|(?:
CharacterData
|Subtree)
Modified|A
(?:
ttrModified
|ctivate)|
Focus(?:
Out|In)|
MouseScroll
)|r(?:e(?:
s(?:u(?:m
(?:ing|e)|
lt)|ize|et
)|
adystatechange
|pea(?:
tEven)?t|m
```

```
                    ovetrack|trieving|ceived)|ow(?:s(?:inserted|
                                                    delete)|e
                                                    (?:nter|
                                                    xit))|
                                                    atechange)
                                                    |p(?:op(?:
                                                    up(?:hid
                                                    (?:den|ing
                                                    )|show(?:
                                                    ing|n))|
                                                    state)|a
                                                    (?:ge(?:
                                                    hide|show)
                                                    |(?:st|us)
                                                    e|int)|ro
                                                    (?:
                                                    pertychange
                                                    |gress)|
                                                    lay(?:ing)
                                                    ?)|t(?:
                                                    ouch(?:(?:
                                                    lea|mo)v
```

```
e|en(?:ter|d)|cancel|start)|ime(?:update|out
)|
ransitionend
|ext)|u(?:
s(?:
erproximity
|
sdreceived
)|p(?:
gradeneeded
|dateready
)|n(?:
derflow|
load))|f
(?:o(?:rm
(?:change|
input)|cus
(?:out|in)
?)|i(?:
lterchange
|nish)|ai
```

```
led)|l(?:o(?:ad(?:e(?:d(?:meta)?data|nd)|
                                                    start)?|
                                                    secapture)
                                                    |
                                                    evelchange
                                                    |y)|g(?:
                                                    amepad
                                                    (?:(?:dis)
                                                    ?connected
                                                    |button(?:
                                                    down|up)|
                                                    axismove)|
                                                    et)|e(?:n
                                                    (?:d(?:
                                                    Event|ed)?
                                                    |abled|ter
                                                    )|rror(?:
                                                    update)?|
                                                    mptied|xit
                                                    )|i(?:cc
                                                    (?:
                                                    cardlock
```

```
                    error|infochange)|n(?:coming|valid|put))|o
                                                        (?:(?:(?:
                                                        ff|n)lin|
                                                        bsolet)e|
                                                        verflow(?:
                                                        changed)?|
                                                        pen)|SVG
                                                        (?:(?:Unl|
                                                        L)oad|
                                                        Resize|
                                                        Scroll|
                                                        Abort|
                                                        Error|Zoom
                                                        )|h(?:e(?:
                                                        adphoneschange
                                                        |l[dp])|
                                                        ashchange|
                                                        olding)|v
                                                        (?:o(?:lum
                                                        |ic)e|
                                                        ersion)c
                    hange|w(?:a(?:it|rn)ing|heel)|key(?:press|
                                                        down|up)|
                                                        (?:
                                                        AppComman|
                                                        Loa)d|no
                                                        (?:update|
                                                        match)|
                                                        Request|
                                                        zoom))[\s\
                                                        0]*=",
                                                        request.
                                                        text))
                if XSSRegex != 0:
                    self.errMsg.setWindowTitle("Vulnerabilities
                                                        Found!")
```

```
                    self.errMsg.setText("We discovered "+ str(
                                                    XSSRegex) +"
                                                    vulnerabilities
                                                     with the
                                                    provided list
                                                    !\nPlease call
                                                     074030495982
                                                    for support!")
                    self.errMsg.setIcon(QtWidgets.QMessageBox.
                                                    Critical)
                    box = self.errMsg.exec_()
        return 0


class BinaryWindow(QtWidgets.QWidget):
    def __init__(self):
        super().__init__()
        self.setWindowTitle("Binary Testing")
        self.resize(1280,480)
        self.layout = QtWidgets.QVBoxLayout()

        # file selector code
        self.filebtn = QtWidgets.QPushButton("Select file to test")
        self.filebtn.setGeometry(QtCore.QRect(270, 90, 80, 23))
        self.filebtn.clicked.connect(self.getFile)
        self.filePathBox = QtWidgets.QTextEdit(self)
        self.filePathBox.setGeometry(QtCore.QRect(160, 50, 311, 31))
        self.filePathBox.setObjectName("filePathBox")
        self.layout.addWidget(self.filebtn)

        self.series = QtCharts.QPieSeries()
        self.chart = QtCharts.QChart()
        self.chartView = QtCharts.QChartView()

        self.buttonBox = QtWidgets.QDialogButtonBox(self)
        self.buttonBox.setGeometry(QtCore.QRect(10, 280, 621, 32))
        self.buttonBox.setOrientation(QtCore.Qt.Horizontal)
        self.buttonBox.setStandardButtons(QtWidgets.QDialogButtonBox
                                                .Ok)
        self.buttonBox.setObjectName("buttonBox")

        self.buttonBox.accepted.connect(self.accept)
```

```
        self.layout.addWidget(self.filebtn)
        self.layout.addWidget(self.filePathBox)
        self.layout.addWidget(self.buttonBox)
        self.setLayout(self.layout)

    def accept(self):
        BinaryWindow.malwareScan(self)

    def malwareScan(self):
        filename = self.filePathBox.toPlainText()
        with Virustotal(API_KEY="
                                            bd213c4441156be78093d19411413f256090c0
                                            ", API_VERSION="v3") as
                                            vtotal:
            with open(filename, "rb") as f:
                bytes = f.read()
                shaHash = hashlib.sha256(bytes).hexdigest()
                resp = vtotal.request(f"files/{shaHash}")
                pprint(resp.data)
                pprint(resp.data["attributes"]["last_analysis_stats"
                                                ])
                self.filePathBox.setText("\nMalicious Ratings: " +
                                                str(resp.data["
                                                attributes"]["
                                                last_analysis_stats
                                                "]["malicious"]) +
                                                 "\nSHA256: "+str(
                                                resp.data["
                                                attributes"]["
                                                sha256"]))
                self.series.append("Malicious",int(resp.data["
                                                attributes"]["
                                                last_analysis_stats
                                                "]["malicious"]))
                self.series.append("Suspicious",int(resp.data["
                                                attributes"]["
                                                last_analysis_stats
                                                "]["suspicious"]))
```

```
                self.series.append("Harmless",int(resp.data["
                                                    attributes"]["
                                                    last_analysis_stats
                                                    "]["harmless"]))
                self.series.append("Undetected",int(resp.data["
                                                    attributes"]["
                                                    last_analysis_stats
                                                    "]["undetected"]))
                self.chartView.chart().addSeries(self.series)
                self.chartView.chart().createDefaultAxes()
                self.chartView.show()


                return 0

    def getFile(self):
        fname = QtWidgets.QFileDialog.getOpenFileName(self, "Open
                                            File",
        "c:\\")[0]
        self.filePathBox.setText(fname)
        return fname


class CreditWindow(QtWidgets.QWidget):
    def __init__(self):
        super().__init__()
        self.setWindowTitle("Credits")
        self.resize(800,800)
        layout = QtWidgets.QVBoxLayout()
        self.setLayout(layout)


class Window(QtWidgets.QWidget):
    def __init__(self, parent=None):
        super(Window, self).__init__(parent)
        self.setWindowTitle("Welcome")
        self.resize(270,110)
        self.WebButton = QtWidgets.QPushButton("Web App Testing")
        self.BinaryButton = QtWidgets.QPushButton("Binary Testing")
        self.CreditButton = QtWidgets.QPushButton("Credits")
        self.UpdateButton = QtWidgets.QPushButton("Update")
```

```
        self.layout = QtWidgets.QVBoxLayout()
        self.layout.addWidget(self.WebButton)
        self.layout.addWidget(self.BinaryButton)
        self.layout.addWidget(self.CreditButton)
        self.layout.addWidget(self.UpdateButton)
        self.setLayout(self.layout)

        self.WebButton.clicked.connect(self.web)
        self.BinaryButton.clicked.connect(self.binary)
        self.CreditButton.clicked.connect(self.credit)
        self.UpdateButton.clicked.connect(self.update)

    def binary(self):
        self.w = BinaryWindow()
        self.w.show()
        self.hide()

    def web(self):
        self.w = WebAppWindow()
        self.w.show()
        self.hide()

    def credit(self):
        self.w = CreditWindow()
        self.w.show()
        self.hide()

    def update(self):
        return 0


def setupUI():
    app = QtWidgets.QApplication(sys.argv)
    window = Window()
    window.show()
    sys.exit(app.exec_())
```

request.py

```
    # This file contains the function that makes requests on behalf
                                    of the application,
                                    essentially a wrapper around
                                    requests.
```

```
import requests as rq

def makeGetRequest(url, params, cookies, auth, hdrs, timeout):

    craftedUrl = url + params

    req = rq.get(craftedUrl, headers, cookies, auth, hdrs)

    return req


def makePOSTRequest(url, data, cookies, auth, timeout, hdrs):
    craftedUrl = url + params

    req = rq.post(url,data=data)

    return req
```

payloads.txt

```
<svg onload=alert(1)>
"><svg onload=alert(1)//
"onmouseover=alert(1)//
"autofocus/onfocus=alert(1)//
```

ApacheRCE.json

```
[
{
    "name": "Apache Flink Unauth RCE",
    "author": "0x0luke",
    "severity": "critical",
    "RequestType": "POST",
    "RequestURL": "/jars/upload",
    "RequestBody": "--8ce4b16b22b58894aa86c421e8759df3\\nContent
                                    -Disposition: form-data;
                                    name=%22jarfile%22;
                                    filename=%22poc.jar%22\\
                                    nContent-Type:application/
                                    octet-stream\\nwarlord\\n
                                    --
                                    8ce4b16b22b58894aa86c421e8759df3
                                    --",
```

```
        "ResponseStatus": "200",
        "Matcher": "success"
    }
]
```