National College of Ireland

22/12/2023

Repeat Submission 05/08/2024

2023/2024

Gabriel Tkacov

X20448246

X20448246@student.ncirl.ie

Computing (Cybersecurity)

Repeat

Vulnerability Scanner and Reporting Tool

Technical Report

Contents

Executive Summary

Max 300 words. Summarise the key points of the report. Restate the purpose of the report, highlight the major points of the report, and describe any results, conclusions, or recommendations from the report.

This report highlights a variety of different aspects to the web application that I have created starting with some expectations and background to the web application such as why I have done this and my reasoning behind it. I will then move onto some of the requirements that are needed to be able to create it following some information about some of the technologies used. Moving on from that you will be presented with a use case of one of the functionalities within the application explaining the process of the task with a result and an alternative example if the user was to make a mistake. Next up you'll receive a further explained list of requirements such as data requirements, user requirements etc briefing you on how the application should act. Lastly there is a bit of discussion behind some of the code itself with screenshots and explanations of what certain parts do.

1.0 Introduction

1.1. Background

I am undertaking this project primarily for my final year project but the reason behind as to why I specifically chose to work on the vulnerability scanner and reporting tool as it is somewhat similar to what I used to work with during my internship last year.

During that period, I was working with different software's where I would create the XML scripts for devices which had a built-in detector that would inform of potential problems that could occur with what was inputted. After the device is configured and used we would then have a reporting tool that would report back a variety of information such as performance and any anomalies.

1.2. Aims

The vulnerability scanner and reporting tool will be set out to detect any anomalies within a web application or database and be able to convert this into a report for a user to read. I hope to achieve a user-friendly application that is efficient and accurate that will provide a detailed report outlining any vulnerabilities that can be found within the system which would save time and resources for a consumer of the product.

1.3. Technology

When creating the application itself I will be mainly using python to create it as it is a very versatile language with a highly extensive library. After doing some research I found

a variety of libraries that I can use that link well with the project idea such as, Nmap which is used for open port identification, Beautiful Soup which is for processing HTML and XML data, Requests which facilitates HTTP requests to web applications and checks for vulnerabilities, ZapV2 to be able to use ZAP on my web application using an API and Flask which is for creating web applications using python. Each of these libraries is exactly the resources I need to proceed with this project.

Regarding algorithms that I plan on using for this I have found an extensive amount that I can use for example. Port scanning which is port scanning techniques used on targeted systems. Scanning scheduling which would be used to schedule a scan during non-peak traffic times which would reduce the time it takes for a scan to complete. Report generation which would work closely with the schedule as it would generate the report straight after the scan and lastly service listing which would identify what services are running on open ports.

2.0 System

2.1. Requirements

The requirements for this application can be extensive for example: 1.When considering my programming language, I went with Python as it is the best option when it comes to creating a vulnerability scanner. For picking a framework I used Flask1.1.2 as it goes well with python and used Beautifulsoup4 as my library for the scanner to be able to access a HTML webpage and do the scanning tasks that I instruct it to. Previously I had used a database and cloud hosting service for this application but decided to remove it as I believe it was causing a clash between other functionalities. In terms of version control I am using GitHub to record any new input/changes. Within the home page of the application there will also be a disclaimer to inform users that this tool should only be used for ethical reasons.

2.1.1. Functional Requirements

Vulnerability scanning should allow the user to enter a URL within the specified field and scan for the vulnerability that the user selects (e.g XSS).

Scan Reporting should display a report of the scan and print out a report if there any vulnerabilities and how many.

The webpage should have a user-friendly user interface that is fast and provides users with a consistent experience.

2.1.1.1. Description & Priority

A description of the requirement and its priority. Describes how essential this requirement is to the overall system.

This Requirement is highly important and is a high priority requirement as it is one of the core functionalities within the application. It allows a user to initiate a scan for cross site scripting of the URL that they insert and submit.

2.1.1.2. Use Case

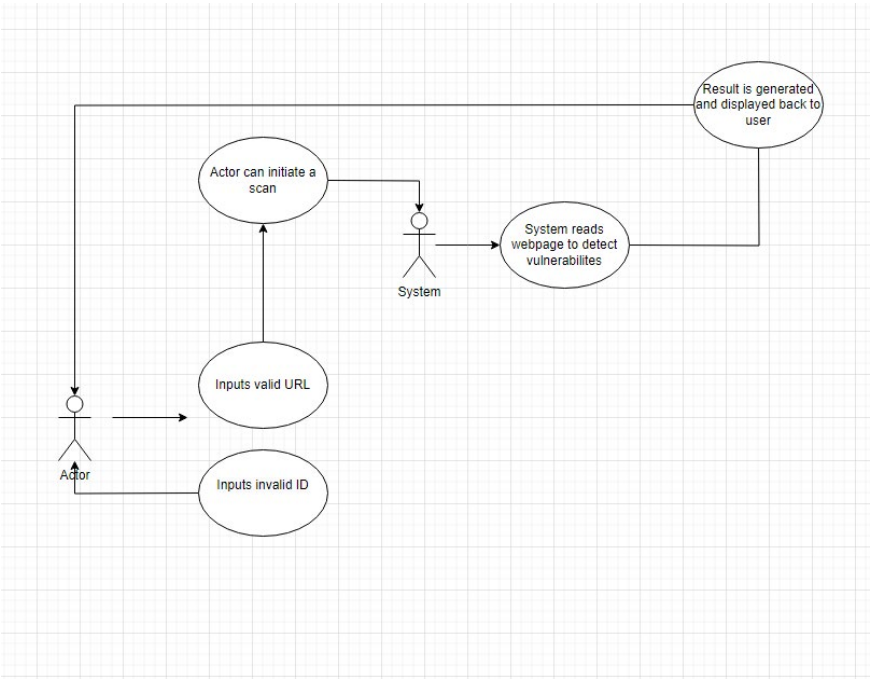
Scope

The scope of this use case is to allow the user to start a automated scan for XSS vulnerabilities within a web page.

Description

This use case describes the process of a user initiating a scan for cross site scripting through the user interface.

Use Case Diagram



Flow Description

Precondition

The system is in operational, and the user is logs in and accesses the user interface

Activation

This use case starts when an <Actor> inserts a URL into the specified field and selects the “scan button”

Main flow

1. The system validates the URL format

2. The <Actor> selects the scan button to start the scan (See A1)
3. The system scans the webpage
4. The <Actor> receives a report of the scan.

Alternate flow

A1 : Incorrect URL Format

1. The system detects an invalid URL and prompts an error message
2. The <Actor> corrects the URL and retries the scan
3. The use case continues at position 3 of the main flow

Exceptional flow

E1 : System Overflow

4. The system tries to run multiple scans at once
5. The <Actor> notices the application does not allow it
6. The use case continues at position 1 of the main flow

Termination

The scan is successful and the system stars reading the URL

Post condition

The system goes into a state where it is ready for a new scan request.

List further functional requirements here, using the same structure as for Requirement1.

2.1.2. Data Requirements

2.1.3. User Requirements

The user requirements focus on the web application to be simple, fast, and secure. As this tool is complex on the backend it is a must that the user interface is made so the user cannot make a mistake. The app must also deliver quick and accurate reports to meet users' expectations.

2.1.4. Environmental Requirements

The environmental requirements for the vulnerability scanning application should have a stable server, regarding this application it shouldn't need to use much of a computer's power to run scans when ran locally. The application should also be compatible with different operating systems and web browser versions.

2.1.5. Usability Requirements

The web application should give clear feedback especially when it comes to generating reports and recommendations. Responsiveness should also be on the high end as it needs to be seen as smooth.

2.2. Design & Architecture

The application is made as a client-server model where the user interacts with a web frontend and data is handled in the backend with scanning operations. The frontend is built using HTML, bootstrap, CSS, and it manages the user interface, input validation and displays results. On the backend it is developed using python with flask and a variety of libraries, it is responsible for the scan requests and scanning algorithms. The main

algorithm at the moment is the XSS scanning algorithm where the scanner would identify common payloads and compare it to the web application to be able to detect for cross site scripting.

2.3. Implementation

This is one of my favourite functions that I have implemented so far. What this does is using an API key from ZAP I am able to run an active scan in ZAP through my web application where it will print out a variety of information such as any alerts that appear while also supplying a status bar to see when your scan will be complete.

```
def zap_scan(target):
    proxies = {
        'http': 'http://localhost:8080',
        'https': 'http://localhost:8080'
    }

    # Disable SSL verification warning
    requests.packages.urllib3.disable_warnings()

    try:
        zap.urlopen(target)
        scan_id = zap.ascan.scan(target)

        print(f"Initiated scan with ID: {scan_id}")

        # To check and see if the scan ID is valid
        if scan_id == 'does_not_exist':
            print("Scan initiation failed: Invalid scan ID returned.")
            socketio.emit('scan_error', {'error': 'Scan initiation failed: Invalid scan ID returned.'})
            return

        while True:
            status = zap.ascan.status(scan_id)
            print(f"Current scan status: {status}")
```

```
            # Had to change the status to integer to fix connection error
            try:
                status_int = int(status)
                if status_int >= 100:
                    break
                socketio.emit('scan_progress', {'progress': status_int})
            except ValueError:
                print(f"Received invalid scan status: {status}")
                socketio.emit('scan_error', {'error': f"Invalid scan status: {status}"})
                return

            time.sleep(5)

        alerts = zap.core.alerts(baseurl=target)
        print(f"Scan complete. Alerts: {alerts}")
        socketio.emit('scan_complete', {'alerts': alerts})

    except Exception as e:
        print(f"Error during scan: {str(e)}")
        socketio.emit('scan_error', {'error': str(e)})
```

Within the XSS scanner this is an example of how it detects for vulnerabilities, a common type of XSS is by creating a script within a field to make it display something and what my scanner does it attempt to find if it is possible to do such on other webpages.

```
        # Had to change the status to integer to fix connection error
        try:
            status_int = int(status)
            if status_int >= 100:
                break
            socketio.emit('scan_progress', {'progress': status_int})
        except ValueError:
            print(f"Received invalid scan status: {status}")
            socketio.emit('scan_error', {'error': f"Invalid scan status: {status}"})
            return

        time.sleep(5)

    alerts = zap.core.alerts(baseurl=target)
    print(f"Scan complete. Alerts: {alerts}")
    socketio.emit('scan_complete', {'alerts': alerts})

except Exception as e:
    print(f"Error during scan: {str(e)}")
    socketio.emit('scan_error', {'error': str(e)})
```

In this snippet you can see this is my navbar that was created using bootstrap. Also take notes of the href's for tabs within the navbar as they have to be routed rather than your usual example.html.

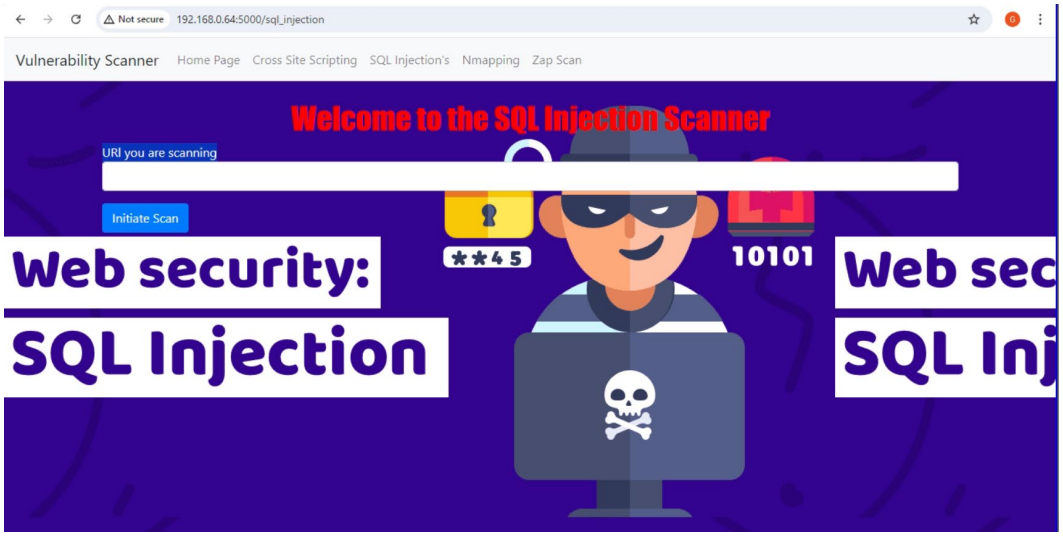
```
<!DOCTYPE html>
<html lang=""en">
<head>
    <meta charset="UTF-8">
    <meta name=""viewport" content="width=devide-width, initial-scale==1.0">
    <title>Vulnerability Scanner</title>
    <link rel="stylesheet" href="{{ url_for('static', filename='style.css') }}">
    <link rel="stylesheet" href="https://stackpath.bootstrapcdn.com/bootstrap/4.3.1/css/bootstrap.min.css">
</head>
    <!-- Navbar being implemented-->
    <body class="{{ body_class }}">
        <nav class="navbar navbar-expand-lg navbar-light bg-light">
            <a class="navbar-brand" href="/">Vulnerability Scanner</a>
            <div class="collapse navbar-collapse" id="navbarNav">
                <ul class="navbar-nav">
                    <li class="nav-item">
                        <a class="nav-link" href="/">Home Page</a>
                    </li>
                    <li class="nav-item">
                        <a class="nav-link" href="/xss">Cross Site Scripting</a>
                    </li>
                    <li class="nav-item">
                        <a class="nav-link" href="/sql_injection">SQL Injection's</a>
                    </li>
                    <li class="nav-item">
                        <a class="nav-link" href="/nmap">Nmapping</a>
                    </li>
                    <li class="nav-item">
                        <a class="nav-link" href="/zap_scan">Zap Scan</a>
                    </li>

                </ul>
```

2.4. Graphical User Interface (GUI)

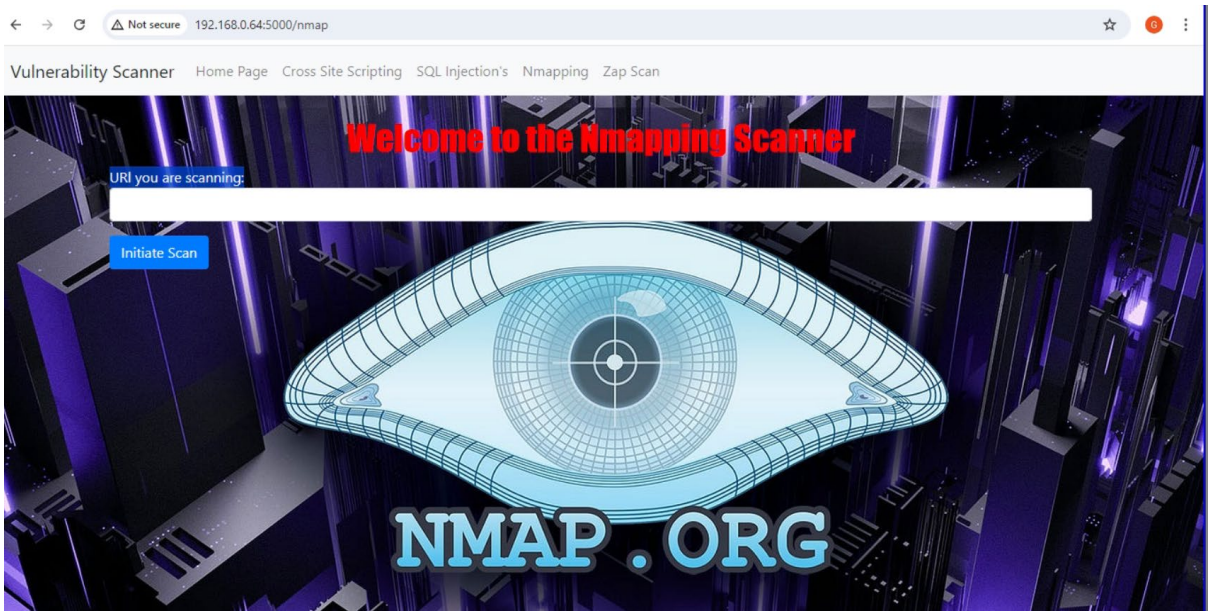
Index.html

Xss.html

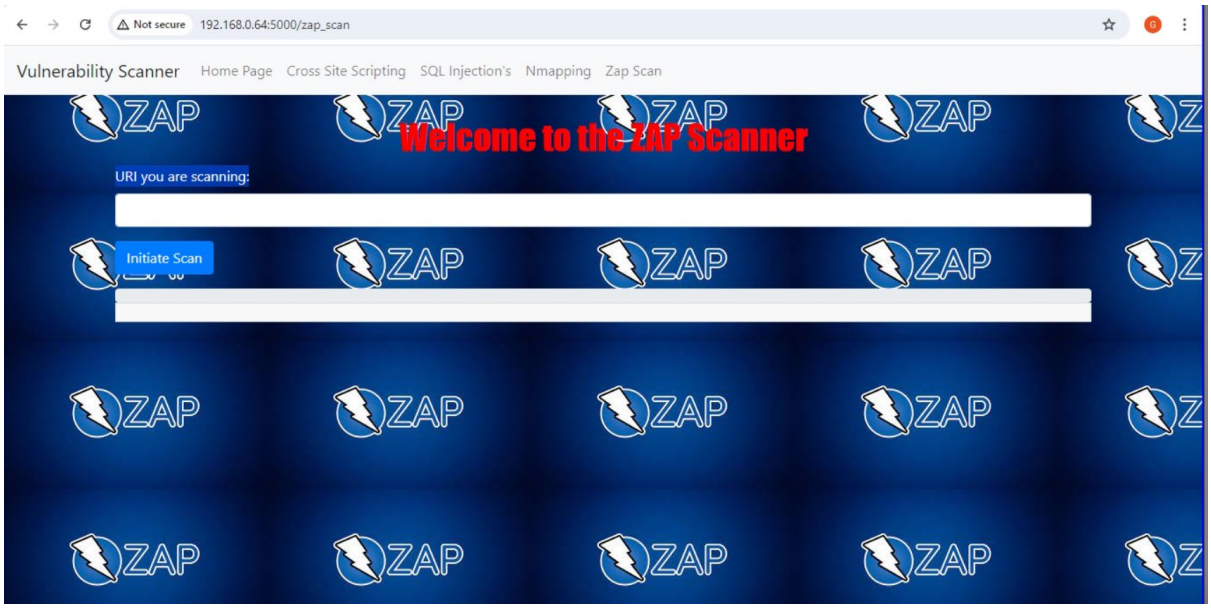
Sql_injection.html



Nmap.html



Zap_scan.html



This is the GUI of the application. This includes fiver different pages that each have their own unique trait. Each tab corresponds to a different vulnerability that you can scan for starting with cross-site scripting then SQL injection's and so on.

2.5. Testing

Throughout this project I had done some of the testing but not as many I would have liked. During the creation of the zap scanner, I was getting issues with connecting ZAP to my web application but after doing some of my own testing with a new project I was able to figure out the issue where I was having SSL certificate issues and this was found by running a python script test. Unit testing was performed during the creation of the application where I had tested every function such as the SQL injection scanner, XSS scanner, Nmap etc. When conducting integration testing I was making sure that every aspect of the project was compatible with each other, this was done by making sure that

my flask application could handle having each of these scanners in one place and making sure nothing breaks down.

3.0 Conclusions

To conclude this report on the vulnerability scanner there are a variety of advantages, disadvantages, strengths, and limitations that I must discuss. Starting off with the advantages I believe if all went well this tool can be a great start for developers out there to know what may be going on in the background of their own websites as this tool covers some of the most threating vulnerabilities out there meaning that this tool can be used extensively for security reasons. Second advantage to this is that it is automated, the web application can be run at any time without anyone needing to do anything except the user himself leading the application to be prone from human errors. Lastly I believe that the most important advantage to this project is that the educational value that comes from it can be used by anyone to learn what is out there and how it can affect them. Moving onto the disadvantages there are some that can be clearly seen, for example new vulnerabilities are always being created meaning that the application would need constant updating where new scanners would need to be added specific to the new vulnerability. Secondly depending on the webpage, false positives can be detected where when a scan is running it could potentially identify something that isn't a vulnerability and cause alerts for no reason. Strengths of this application occur with the likes of saleability whereas mentioned before if new vulnerabilities are created it can easily be added to the web application without damaging the existing scanners. Secondly the user interface of this application is incredibly user friendly where everything is laid out in a way for users to be able to find anything they need such as the specific test they would like to run or any information that they might need. Lastly moving onto the limitations of this project starting with the main one which is my own ability in the python language. Generally, the application works fine but there is always room for improvement, and I will be making sure to keep going python as I am finding it a lot more interesting than other languages out there. The next limitation at the minute is that the application only covers the most common vulnerabilities meaning that it may only be accustomed to certain users. In conclusion this project can hold significant positives from it but also contains a lot of challenging aspects which I was unfortunately not capable of.

4.0 Further Development or Research

With additional time and resources, which direction would this project take?

If I had additional time and resources with this project I would like to make two sperate versions of the application where one of them can be a lite version which is specific for educational purposes to teach others about the threats and dangers with some of the vulnerabilities. Also, within this application I would allow for small exercises to demonstrate how such vulnerabilities can be created. Moving onto the other version it would be expanded to an all-in-one application from the original. My thinking behind this would be to include the original vulnerabilities that I mentioned but expand on it to include the OWASP

top 10 vulnerabilities so that the application would scan for the most threating vulnerabilities out there.

5.0 References

Please include references throughout your document where appropriate. See here for a guide on referencing from the NCI library.

Pykes, K. (2022) *How to use pytest for unit testing*, *DataCamp*. Available at: https://www.datacamp.com/tutorial/pytest-tutorial-a-hands-on-guide-to-unit-testing (Accessed: 12 May 2024).

Abdeladim Fadheli, M.A. (2019) *How to build an XSS vulnerability scanner in python - the python code*, *Python Code*. Available at: https://thepythoncode.com/article/make-a-xss-vulnerability-scanner-in-python (Accessed: 12 May 2024).

Owasp Top Ten (no date) *OWASP Top Ten | OWASP Foundation*. Available at: https://owasp.org/www-project-top-ten/ (Accessed: 12 May 2024).

Mark Otto, J.T. (no date) *Navbar*, · *Bootstrap v5.3*. Available at: https://getbootstrap.com/docs/5.3/components/navbar/ (Accessed: 12 May 2024).

Bona, T. (2023) *Building an XSS scanner with python: Detecting cross-site scripting vulnerabilities - by Tommaso...*, *Medium*. Available at: https://systemweakness.com/building-an-xss-scanner-with-python-detecting-cross-site-scripting-vulnerabilities-by-tommaso-69d4c9e04d72 (Accessed: 12 May 2024).

Meet the Author Pablo Zurro Cybersecurity Product Manager Core Security and Profile, V. (no date) *Top 14 vulnerability scanners for cybersecurity professionals*, *Top 14 Vulnerability Scanners for Cybersecurity Professionals | Core Security Blog*. Available at: https://www.coresecurity.com/blog/top-14-vulnerability-scanners-cybersecurity-professionals (Accessed: 12 May 2024).

6.0 Appendices

This section should contain information that is supplementary to the main body of the report.

6.1. Project Proposal

National College of Ireland

Project Proposal

Vulnerability Scanner and Reporting Tool

20/10/2023

Computing

Cybersecurity

2023/2024

Gabriel Tkacov

X20448246

X20448246@student.ncirl.ie

Contents

1.0 Objectives

(Max half Page)

What does this project set out to achieve?

The vulnerability scanner and reporting tool will be set out to detect any anomalies within a web application or database and be able to convert this into a report for a user to read. I hope to achieve a user-friendly application that is efficient and accurate that will provide a detailed report outlining any vulnerabilities that can be found within the system which would save time and resources for a consumer of the product.

2.0 Background

(Max half Page)

Why did you choose to undertake this project? How will you meet the objectives set out in Section 1.0?

I am undertaking this project primarily for my final year project but the reason behind as to why I specifically chose to work on the vulnerability scanner and reporting tool as it is somewhat similar to what I used to work with during my internship last year.

During that period, I was working with different software's where I would create the XML scripts for devices which had a built-in detector that would inform of potential problems that could occur with what was inputted. After the device is configured and used we would then have a reporting tool that would report back a variety of information such as performance and any anomalies.

3.0 State of the Art

(Max half page)

What similar applications exist already? What makes your project stand out? How does it differ from similar work of others?

There are a variety of similar applications that already exist for example there is a tool called Acunetix which is a well-known vulnerability scanner that identifies security vulnerabilities in web applications like SQL injections and cross site scripting. As well as that there is also Qualys which is a cloud based vulnerability scanner that does automated reporting for web applications and networks.

What makes my project stand out compared to other competitors is firstly simplicity, as I am still at a intermediate level within my studies and technological knowledge the application

that I will be making a easy to use and simple looking GUI to not complicate anything for a end user. Customisation will also be something that would stand out in my opinion which I will discuss further in the documentation. To put it in simple terms I believe my application will be easy to use, customizable to the user's needs and the application itself is for educational purposes as it will teach others what the application is about rather than just using it like a corporation would do.

4.0 Technical Approach

(Max 1 page)

What approach will you take to development? How will you identify requirements? How will you break down requirements into project tasks, activities and milestones?

To begin development, I will first consult a variety of people such as, my project supervisor, old co-workers that I worked with during my internship and possibly some other connections that I made to seek advice at how I should proceed with the project. I will then look into similar made applications and use it as a guide to help me progress and find different aspects that I wouldn't have imagined adding to my own program. During this process I will be writing down all the requirements needed for my final application and then slowly start branching them off into their separate tasks and activities. For example, a task involved with the requirement of the vulnerability scanner would be coding in a "Back" button.

When talking about keeping track of my progress and milestones I will be doing this using a Gantt chart that I have created which I will be constantly updating with new deadlines and objectives for the whole process of the project including not just coding implementations but also the likes of the final report and video presentations that need to be completed. An example of some of the milestones would be, the completion of the scanning engine, the design of the user interface, development of the database etc.

5.0 Technical Details

(Max 1 page)

Implementation language and principal libraries. What are the important algorithms or approaches under consideration for this work?

When creating the application itself I will be mainly using python to create it as it is a very versatile language with a highly extensive library. After doing some research I found a variety of libraries that I can use that link well with the project idea such as, Nmap which is used for open port identification, Beautiful Soup which is for processing HTML and XML data, Requests which facilitates HTTP requests to web applications and checks for vulnerabilities, SQL Alchemy which can be used to manage SQL databases using the python language and Flask which is for creating web applications using python. Each of these libraries is exactly the resources I need to proceed with this project.

Regarding algorithms that I plan on using for this I have found an extensive amount that I can use for example. Port scanning which is port scanning techniques used on targeted systems. Scanning scheduling which would be used to schedule a scan during non-peak traffic times which would reduce the time it takes for a scan to complete. Report generation which would work closely with the schedule as it would generate the report straight after the scan and lastly service listing which would identify what services are running on open ports.

6.0 Special Resources Required

(Max half page)

What special resources, if any will be required for this work?

At the time of creating this I don't believe there are any "special" resources that are needed except improvements in my knowledge of python where I will spend my time studying and enhancing my skills. Another piece that I will need to consider is to ensure I am compliant with the college's rules and regulations, especially when dealing with scanning networks. Lastly, I may need to use virtualization tools such as virtual machines to have everything in one controlled space which would also be great to use during the time of testing.

7.0 Project Plan

(Max 2 pages)

Project plan with details on implementation steps and timelines. This project plan should provide as much detail as possible for now and will be revised with more detail with the mid point documentation.

As of right now I do not have a full list of the exact dates and timelines of my outcomes, but I have an estimate of how long each requirement will take. Starting off with the deliverables for the project itself the dates of these are saved on Moodle and will be worked on within the time periods given. In terms of the actual program, I believe one week to prepare myself with downloading the necessary software and tools to begin this project such as python extensions and libraries on Visual Studio Code.

Within this week I would also do more research on the types of vulnerabilities out there and expand my knowledge behind my anticipated outcome. The next couple of weeks coming up to the midpoint presentation I would spend the rest of the time getting my prototype created to demonstrate how the application would roughly run to demonstrate its capability.

After this I will be fully focusing on the progression of the program where I am estimating a time period of about 10-12 weeks of development where I would like to have my scanning

engine complete, user interface to be fully integrated and the functionalities of the reporting tool are working correctly.

Continuing on from that I would then give myself a max of a week or two for the testing phase of the application where I would try scope out any bugs that need fixing and have users testing out the application with a feedback form to give back after using the application. Once this is completed and I am happy with the results I would then give myself about a week to deploy this live.

Lastly I would then give myself about 4-6 weeks on creating the documentation for this and a video presentation of how the program should be used with test cases that can be used.

A Gantt chart will be created documenting all of these steps and more within the week.

8.0 Testing

(Max 1 page)

Describe how you will evaluate the system with real technical data using system tests, integration tests etc. If applicable describe how you will evaluate the system with an **end user. (be careful here re Ethics etc)}**

A vital part when it comes to creating an application is to make sure that needed tests are done to thoroughly check if your application is usable, reliable and stable.

The testing that I would do for my application would involve the likes of unit testing different components to verify that functions are operating correctly, integration testing to ensure different parts of the code work with each other (e.g ensuring that the reporting tool works with the scanner itself).

Performance testing would be taken into consideration to test the rate at which the application is running and making sure that it is good enough for a end user.

Lastly in terms of system testing I would also do some robustness testing to see how the application handles information that doesn't follow the applications conditions.

After the system tests I would like to do some end user testing where I would ensure ethical considerations are taken into account where I would ask for consent from people to try the application and leave some feedback. Including this I would also make sure not to scan systems without their permission.

8.1. Reflective Journals

Gabriel Tkacov x20448246

Cybersecurity Specialisation

From the beginning of the year up until the end of this month not much has been done in terms of this project except the following. Starting off we were asked to come up with a project pitch which was a three-minute-long video describing my idea and sending it off for approval. I came up with the idea of web application vulnerability tool that will be able to detect any vulnerabilities of the application and generate a reporting listing all of the issues. After sending this on and getting an approval we were then asked to write up our project proposal which was document outlining the grasp of the project and what I hope to achieve in the end. Lastly we received our project supervisors, I was assigned to Keith Maycock who I will be keeping in touch with to receive guidance and advice on my future ideas. That is the current progress of this month.