Unified Vulnerability Scanning Engine and Management System

Project Specification and Plan

Andrew Gilbey C00263656

Supervisor: Richard Butler

SETU – Carlow

Table of Contents

Table Of Figures

1. Introduction

1.1 Background

In today's digital landscape where vast amounts of data are transferred across and stored using the internet, and with the sophistication of cyberattacks becoming greater, the

necessity of cybersecurity cannot be overstated because it stands as the first line of defence against these cyberattacks and so can protect businesses against data breaches, financial losses and damage to company reputation, (Lo-A-Njoe, C, 2023), and mitigate against the growing harm cyberattacks pose to customers and society, (Bada & Nurse, 2019). Therefore, vulnerability assessments have gained a critical importance as they help organisations identify vulnerabilities in their systems before they can be exploited, (IARM, 2023).

This project aims to develop an integrated and automated vulnerability scanning system to better meet the cybersecurity needs of organisations and by focusing on both efficiency and accessibility, seeks to simplify the vulnerability assessment process for organisations of all sizes.

1.2 Problem Statement

While various vulnerability scanning tools are available, they frequently operate in isolated silos, making vulnerability management cumbersome, (Saxena, 2023). These systems will often require manual intervention for result analysis which can in turn lead to inconsistent findings and the potential for overlooked or missed vulnerabilities. This lack of integration not only consumes valuable time and resources but also leaves room for human error, compromising the reliability of the assessments, (Smith, 2022).

Moreover, without the presence of an effective benchmarking that can provide continuous comparative reports, businesses may struggle to accurately monitor their cybersecurity performance over time and this then would make it difficult to identify which areas of their cybersecurity have improved or regressed resulting in a substandard cybersecurity strategy, (Smith, 2022).

Given these challenges, there is a need for an integrated solution that goes beyond just the automation of the vulnerability scanning process. The proposed system will incorporate a comparative analysis feature that sets it apart from existing solutions.

Comparative analysis in this context means the system will not only identify vulnerabilities but also compare them against historical data/previous scans. This feature provides a multi-dimensional view of an organisation's cybersecurity strategy over time, rather than one snapshot of vulnerabilities at a single point in time, meaning It would enable organisations to measure their security improvements or regressions, aiding in informed decisions based

on trends rather than isolated incidents. This comparative analysis can extend over an extended period of time, such as six months or a year, allowing organisations to evaluate their cybersecurity over a longer time period and so better appreciate the impact of their security investments, spot seasonal vulnerability patterns and enabling them to make data-driven decisions for continuous improvement in their cybersecurity approach which will protect their long term sustainability, reputation and any potential financial damage, (Lo-A-Njoe, C, 2023), that could occur as well as protect their users/customers, (Bada & Nurse, 2019).

By integrating this comparative analysis utility, this would transform vulnerability management from a reactive process to a proactive strategy and would offer a more enhanced understanding of specific vulnerabilities, thereby significantly enhancing the efficiency of their cybersecurity assessments (Intelligence, 2023).

2. What is an Automated Vulnerability Scanner?

An Automated Vulnerability Scanner is a specialised piece of software designed to autonomously identify security vulnerabilities in an organisation's network. This tool scans various components such as web servers, operating systems, and other end points for any known security weaknesses, (Vulnerability scanner: What is it and how does it work, n.d).

2.1 Integration of Open-Source Tools

This project transforms the traditional concept of an “Automated Vulnerability Scanner” by merging multiple open-source tools, such as Nmap, OpenVAS, ZAP, and DNS resolvers, into a unified security assessment solution. Instead of acting as standalone software, this scanner operates as a collection of integrated unified scanning engine that coordinate these tools to deliver a single comprehensive vulnerability report. By leveraging a multi-layered approach, the scanner would provide a more in-depth analysis than traditional methods, which would streamline the traditionally fragmented vulnerability assessment process by arranging the functionalities of these different tools into one operation. Upon execution, the unified scanning engine will initiate a series of security tests across user selected domains and

networks—from network mapping to web application testing—and collate the findings into a single report. This incorporated reporting makes it easier to assess and prioritise vulnerabilities, thus simplifying the task of automated report generation and vulnerability scanning.

2.2 The Benefits of Unified Reporting and Benchmark Utility

One of the most significant advantages of generating a single report is that it provides a centralised source of data. With all the vulnerability metrics gathered in one place, businesses can achieve a holistic understanding of their security posture and allow insights to empower organisations to make well-informed decisions for timely remediation.

Unified reporting also facilitates better collaboration among cybersecurity teams. In traditional setups where different tools operate in silos, teams often find themselves working disjointedly, leading to inefficiencies and possible communication gaps, (McBee, 2022). This reporting methodology ensures that everyone—be it network analysts, engineers, or system administrators—works from the same dataset, thereby enhancing teamwork and resulting in a more efficient vulnerability management process.

One of the unique features of this project is the built-in benchmark utility. Designed to streamline the assessment process in order to provide more meaningful insights over time, this utility is initialised for all subsequent scans once the initial vulnerability scan is complete and the first report is generated. When a new scan is executed, the utility compares the findings with the previously generated report. The main objective of this feature is to eliminate any redundant information and focus on the changes since the last scan. This can range from newly discovered vulnerabilities and changes in severity level to any successful mitigations that have been applied and so by highlighting these changes, the benchmark utility provides a dynamic view of the security landscape, making it easier to prioritise actions and allocate resources effectively. This benchmarking utility can also quantify the number of vulnerabilities, offering management a metric for assessing the scale of the security risks. This can used be to determine if additional training is required for the cybersecurity team.

In line with ensuring effective vulnerability management, the benchmark utility is also equipped with Service Level Agreements (SLA) tracking capabilities. It does this by comparing the timestamps of when vulnerabilities are discovered and when they are remediated against the timeframes defined SLAs. If a vulnerability is not addressed within the agreed-upon time, the utility flags these instances as potential SLA non-compliance. This constitutes as a critical alert for management, signalling that immediate action is required and perhaps a different approach may be required for effective vulnerability management.

3. Deliverables

This section outlines the project's deliverables, and is categorised into 'core' and secondary.' Core deliverables are essential for achieving the project's main goal, and are further subdivided into 'System-Specific' and 'Non-System Specific' to provide a more nuanced understanding of their significance. In contrast, secondary deliverables enhance the system's functionality but are not critical to its basic operation.

3.1 Core Deliverables

3.1.1 System-Specific Core Deliverables

1. **Unified Vulnerability Scanning Engine and Management System (UVSEAMS)**
   This is the central part of the project that performs comprehensive network and optional web application scans to identify security vulnerabilities. It will leverage the power of open-source tools like Nmap, OpenVAS, and OWASP ZAP to autonomously scan, detect, and report on vulnerabilities within a network or web assets.

2. **Intelligent Reporting Mechanism**
   The Intelligent Reporting Mechanism takes the output from each scan and compares it with previous scan. It highlights any changes in the security posture, identifying new vulnerabilities that have appeared and previous vulnerabilities that have been mitigated. This would prevent security teams sifting through pages of raw data or redundant data.

3. **Exclusion List for Scan Scope**

There may be instances where certain assets should be excluded because they are out of the project's scope. The Exclusion List would allow a user to define these parameters, ensuring that the system's scanning resources are focused on selected assets. This would include a function to include web crawling that would spider through every page of the target web application in order to generate a full list of URLs, these can then be filtered out based on the user-defined exclusion criteria, (Kariyawasam, 2021) in order to be excluded from the subsequent scans, this can be achieved with automating tools such as OWASP ZAP, (Tahalani, 2020).

4. **Database Solution**

The Database Solution is integral not only for data retention but the foundational layer of the report comparison logic. This feature captures all scan outcomes (reports) and, serve as the foundational layer for the benchmark utility. By comparing new scans with stored previous scans, the system highlights any shift in the security landscape, like new vulnerabilities or mitigations, offering a different view.

The database also supports user-based vulnerability management, allowing specific team members to be assigned to particular vulnerabilities for targeted remediation. To enhance the accountability of this process, the system will have implemented a "Retest Required" flag. This flag can be set in order to indicate that a particular vulnerability has been addressed but has not yet been retested so aids in timeline management, allowing teams to prioritise retesting and validation; streamlining the remediation process.

To enhance security of the data, AES-256 Galois/Counter Mode (GCM) encryption will be employed (MSFT, n.d.). This encryption algorithm ensures that the stored scan outcomes and any user data are secure and accessible only to authorised personnel. The use of GCM allows for an efficient and fast encryption/decryption process, (Vocal Tech, n.d.) and so making it a seamless addition to the system

3.1.2 Non-System Specific Core Deliverables

1. **Research Report**

   This report serves as the cornerstone of the entire project, designed to provide an insight into the methodical considerations guiding the project's development. It will outline the chosen methodologies, and be used as a resource for understanding not just the 'what,' but also the 'why' and 'how' of the project. Its primary aim being to offer anyone reviewing it a complete understanding of the thought processes that underline every aspect of the project.

2. **Project Report**

   A summative document that encapsulates the project's journey from its conception to completion. It is designed to reflect the various learning curves, challenges, and setbacks encountered, along with any strategies used to overcome them.

3. **Web Page**

   The project's web page will function as a multifaceted portfolio showcasing the project's milestones and documentation. It will feature a user-friendly interface where visitors can access the projects documentation and find detailed explanations of the projects core functionalities and aims.

4. **Research Poster**

   The research poster will be used to provide a visual summary of the project and will be designed to showcase important information in an easily digestible format. It will highlight the problem statement, outline the methodology used, and present any key findings resulting from the research. This poster will be created with both academic and industry audiences in mind to ensure it is suitable for presentation and serve as a tool for distribution and feedback collection.

3.2 Secondary Deliverables

1. **Automated Email Notifications**

While not essential to the core functioning of the system, automated email notifications serve a critical role in expediting response times. Using Python's SMTP libraries, this feature would allow for the system to automatically send reports of only high-level vulnerabilities to designated recipients. By focusing only on high-priority issues, these real-time email alerts will ensure that inboxes are not burdened with excessive notifications, while still making certain that the right people are informed immediately, and so enhance the operational effectiveness.

2. **Flexible Scheduling Feature**

   This feature allows users to set up vulnerability scans at intervals of their choosing. Whether it's a one-off scan or recurring scans, a flexible scheduling feature ensures that the system can adapt to various operational needs without requiring manual intervention every time a scan is due.

4. Project Functionality

The function of this UVSEAMS will be designed to be wide-ranging, integrating various operational and data management aspects to provide an efficient solution. These functions are not standalone components but are interdependent modules that all work in conjunction with each other to automate vulnerability scanning. Below, are the listed functions are grouped into three categories: Operational, User Interface and Data Management. Each category serving as a specific purpose contributing to the overall effectiveness, usability and function of the system.

4.1 Operational Functions

1. **Asset Discovery and Network Scanning**

   This function integrates two key elements for an overview of the network—Port scanning (Nmap) for network scanning and utilising a Python DNS resolver library for DNS reconnaissance. Nmap is employed to discover active hosts and services and parallelly, the Python DNS resolver is used to gather DNS records. To avoid potential issues such as over-scanning sensitive areas or triggering legal concerns, users have the option to manually configure the subdomains that will be included in the DNS reconnaissance. This information would be able to reveal additional details about how network traffic is routed and uncover

potential security risks such as misconfigured DNS records.

2. **Vulnerability Scanning**

   The system will use a vulnerability scanner, in this case, OpenVAS for general vulnerability scanning across network devices and if selected OWASP ZAP for focused web application vulnerability assessments. These tools provide up-to-date checks against an array of known vulnerabilities.

3. **Data Collection and Formatting**

   The system will utilise an Integrated Unified Scanning Engine developed in Python that act as the middleman between the scanning tools and the reporting engine in order to weave everything together. This Integrated Unified Scanning Engine will be responsible for collecting raw scan data, parsing it, and formatting it into a structure that can be easily interpreted and visualized. This ensures that the data presented in a manner that can be readily acted upon.

4. **Report Generation**

   ReportLab will be integrated into the system for the purpose of converting formatted scan data into detailed, readable reports. These reports are designed to offer different levels of detail tailored to the specific roles within the security team, such as penetration testers or engineers and by utilising different SQL queries for each role, the system generates actionable insights that are most relevant to the responsibilities of each team member. This allows security professionals to better act upon any vulnerabilities that have been detected.

5. **Alerting and Email Notification**

   The system leverages a Python's SMTP library to automate the process of email notification. Once a scan is completed or a significant vulnerability is detected, and

can automatically alert designated recipients via email, ensuring prompt action can be taken.

6. **Trend Analysis and Benchmarking**

   Built-in comparison logic will allow for the correlation of new scan results with the previous report stored in the database. This enables benchmarking, giving security teams the ability to see how their network's security posture has changed over time and cuts down on redundant data.

7. **Task Scheduling**

   By incorporating scheduling tools such as Celery or Cron, the system enables users to set up vulnerability scans at customised intervals. This will ensure that vulnerability assessments become a continuous, rather than a one-off, process and to facilitate trend analysis and benchmarking.

8. **Assigning Vulnerabilities to individuals.**

   The system incorporates a role-based assignment feature that allows for the targeted allocation of identified vulnerabilities to specific team members for remediation. Utilising a user-friendly dashboard (See 9), analysts can assign vulnerabilities to engineers based on their workload. This feature enhances the efficiency of the vulnerability management process by ensuring that each vulnerability is addressed by the most qualified individual. This would aid in accountability, as each team member is clearly responsible for specific tasks, facilitating more effective communication within the team.

9. **Role-Based Dashboard**

   Upon successful login, the system will present users with a specialised Dashboard tailored to their specific role within the organisation—be it a penetration tester,

engineer, analyst, or system administrator. This role-based dashboard serves as the centralised hub for all user activities and is designed to display only the most relevant information for each role. For example, penetration testers will have quick access to initiate new scans and review past results, while engineers will see a list of assigned vulnerabilities that require remediation. Analysts may have an overview of ongoing scans, vulnerabilities, and assignment statuses.

4.2 User Interface Functions

1. **Login & Authentication**

   This feature will employ secure protocols to verify the identity of users, granting them access to the system ensuring that only authorised personnel can interact with the vulnerability assessment tool.

2. **Configurable Settings/Exclusion List**

   This function provides users with the ability to tailor various aspects of the scanning process and report generation. Users can customise parameters such as scan depth, frequency, and types of vulnerabilities to look for, thereby creating a vulnerability assessment process that best suits a specific need and within a specific scope.

3. **Graphical User Interface**

   The system will feature a Python-based which will utilise the tkinter python library which was chosen as it allows for convenient and fast creation of GUIs, (Pedamkar, 2023). Functions such as initiating a scan, setting up parameters, and defining an exclude list will be accessible through dedicated buttons, fields, and combo boxes within specific role-based dashboards. Upon executing a task, the system will compile the results into an external PDF/CSV file, and then store this into the database although will also be accessible directly through the GUI. This GUI approach offers a more user-friendly experience and has a lower skill gap when compared to a CLI system which would require more technical expertise and is far more tedious,

(Adams, 2018).

4. **Admin Interface**

   Upon successful login as an administrator, the system will redirect the user to a specialised admin dashboard. This specific dashboard is designed to centralise all administrative functions, offering a solution for managing user roles, adding or deleting users, and other system configuration. This dashboard will feature sections dedicated to each administrative use case. For example, a 'User Management' panel where admins can add, remove, or edit user roles and permissions. This interface enhances operational efficiency by providing administrators with the tools they need to manage the system effectively, and all in one place.

4.3 Data Management Functions

1. **Data Storage**

   All scan results, reports, and any historical data are securely stored in an encrypted database. This forms the basis for long-term trend analysis and for comparing new scans against past results to identify changes in order to utilise the benchmark utility.

2. **Benchmark Comparison**

   This feature will use built-in comparison logic to contrast new scan results with previous ones. To facilitate this, scan data is retained in a structured database that employs unique identifiers for each vulnerability and timestamps for each scan. Using Python comparison logic, the system will query this database to retrieve relevant historic and present scan results and by comparing these datasets, the system can identify trends, such as security improvements or deteriorations over time.

4.4 System Architecture Diagram

The architecture diagram below provides a visual map of the project's Infrastructure, showing the Python scripts, Docker-hosted SQL database, and the various functions that work together in order for the system to function.

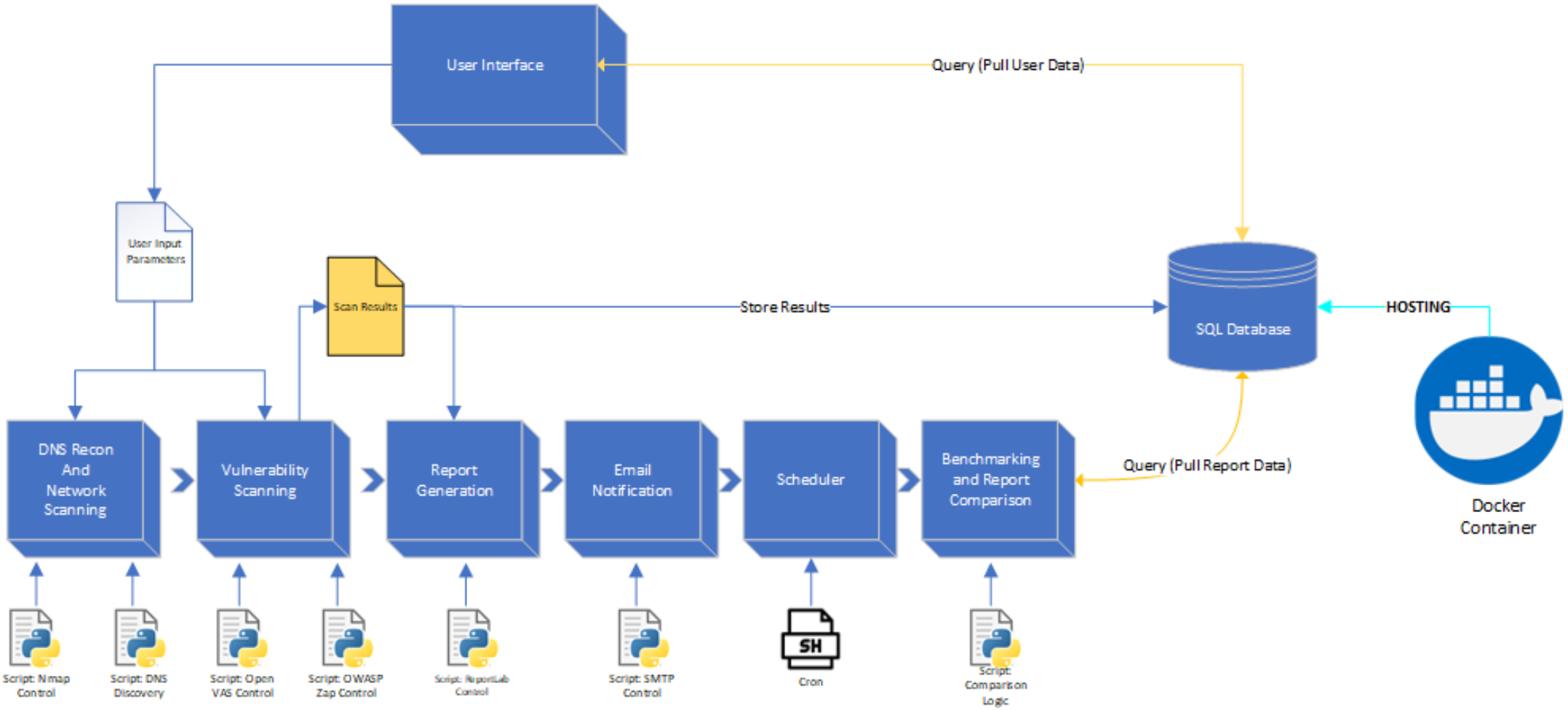

Figure 1: System Architecture Diagram

4.5 Tools and Technologies

4.5.1. Programming language

Python has been selected as the primary programming language for this project for several reasons; It has an easy to write and read syntax when compared to other languages such as Java and C++, Second, Python is open-source, (Jaiswal and Dwivedi, 2021) and as such provides the advantage of a strong community of contributors, this means it has a significant number of libraries, making it versatile, and enabling the integrating various tools that the system is required to use and so would enable the project to utilise existing solutions for tasks like network scanning, vulnerability assessment, and web application testing, (Surve, 2023) reducing the overall time development time.

4.5.2 Tool Selection

This Step involves identifying the open-source tools that will be integrated into the unified scanning engine. A list of all selected tools and a description of their functionalities will follow.

4.5.2.1 Network Scanning and Recon

1. **Nmap (Network Mapper)**

   Nmap has been chosen for its network discovery capabilities and will be an essential module of the system. Its versatility in identifying open ports and services on a network will serve as the initial layer for the system's security assessment, (Artykov, 2021). By integrating Nmap, the aim is to provide an initial scan that identifies network assets, this information will then be used in conjunction with other tools in the system.

2. **OpenVAS**

   OpenVAS has been chosen for its vulnerability scanning capability and is a crucial part of the security assessment framework of the system. OpenVAS specialises in vulnerability assessment and scans network assets for known vulnerabilities, misconfigurations, and other potential security risks and then provides a detailed report of its findings (Das, 2021).

3. **OWASP ZAP**

   OWASP ZAP (Zed Attack Proxy) has been chosen for its specialised focus on web application security. It is designed to find various types of security vulnerabilities within a given web application which will compliment the systems aim to offer web application scanning, (Ansell, 2023). ZAP also offers a web crawl utility which can automatically navigate through a web application to discover its content, (Tahalani, 2020)– This will enable the system to understand the full scope of web assets and assist in systems exclude list functionality. Therefore, integrating OWASP zap the system will be able to offer a more thorough web application security assessment.

4. **DnsPython**

   DnsPython is a Python toolkit that will be included as part of system for its support of DNS reconnaissance. It supports almost all record types, (Dnspython Contributors(a), n.d.) and also offers both high-level and low-level access to DNS, making it a versatile tool as High-level classes in Dnspython enable efficient querying for data by name, type, and class, returning a well-structured answer set while low-level classes allow for the direct manipulation of DNS zones, messages, names, and records. With support for almost all RR types, Dnspython Is a versatile tool for DNS reconnaissance and is an important part of the system's overall toolkit, (Contributors(b), n.d.).

4.5.2.2 Data management Storage

1. **Docker**

   Docker will be utilised in the system to host the SQL database and it works by creating isolated containers that act like mini virtual machines. These containers are defined by images that specify the exact software package needed, and ensure a consistent and reliable environment for the database, (Anglin, n.d.). This makes Docker an efficient solution for hosting the systems SQL database.

2. **SQL Database**

   This system will be using an SQL database for its ubiquity, ease of use, seamless integration capabilities and high performance, (Indeed Editorial Team, 2023). SQL databases offers organised tables for structured data storage, where each row and column represent unique data items and specific information fields respectively, (Solar Winds Team, n.d.). SQL's ability to integrate smoothly with languages like Python enhances the system's overall efficiency and Its high-speed operation will further enables quick data retrieval, (Indeed Editorial Team, 2023), making it an ideal choice for the system.

4.5.2.3 Scheduling, Communication and Reporting

1. **Smtplib**

   Smtplib is a module in Python's standard library that is used for sending emails via the Simple Mail Transfer Protocol, (Tutorials Point, n.d.). The library offers the flexibility to customise email headers, recipient and other email attributes allowing for a tailored email solution. Smtplib is built into pythons' standard library which means there would be no additional requirement for external package which would simplify the development process for the system, (Python Software Foundation, n.d.).

2. **Cron**

   Cron serves as a task scheduler specifically designed for Linux and its primary function is automate tasks at specified intervals. Cron also offers a reliable way to automate python script execution, ensuring that the system performs its functions at the right times without any manual intervention, (Radečić, 2021), this will allow Cron to be utilised by the system for scheduling follow up scans or re-tests.

3. **ReportLab**

   ReportLab offers a flexible solution for generating PDF reports using python. It has a wide range of features from text and graphics to more advanced capabilities like chart generation and graphics – this is while maintaining a high performance. This makes it an ideal choice for the system when generating reports, as it can easily handle complex reporting requirements that are anticipated, (ReportLab, n.d.).

5. User Groups

Small to Medium-sized Enterprises, (SMEs) are often time considered the “lifeblood” an economy, and are considered as such in Ireland for example, (Ibec, n.d.). The target audience for this project is Small to Medium-sized Enterprises (SMEs). SMEs often overlook their cybersecurity solutions, (Get Advanced, n.d.), as they can be expensive and complicated to implement, (Lake Ridge, n.d)

This project aims to bridge that gap by providing an affordable (open-source) and user-friendly solution for vulnerability assessment for this target audience.

While the system would address the needs of larger enterprises successfully, the system’s use of open-source tools makes it a more affordable, and thus, attractive solution for SMEs when compared with current market solutions, (Debar, 2022), as they can often operate within constrained budgets. Its usability and ease of use due to its purposeful design, make it particularly well-suited for SMEs who due to their size and nature, may not have many resources with dedicated skill or knowledge of this space, for example, they may not have a dedicated IT Security team, (Admins, 2023). Therefore, a streamlined, automated, accessible and affordable system such as the Unified Vulnerability Scanning Engine serves to address a fundamental gap in the infrastructure of SMEs in a sustainable way; making it a valuable asset, (Admins, 2023).

In terms of scalability, the architecture of the system is designed to be modular. This enables organisations to scale their security measures in line with their growth, adding new features or modules as their needs evolve, (Clarke, 2023).

6. Use Cases

6.1 Use Case Diagram

This displays a visual representation capturing the functional requirements of the system in order to illustrate the various interactions between different actors. This diagram serves as a guide for the projects system design, offering a simplified although comprehensive view of system functionality, (Fonseca, 2022).

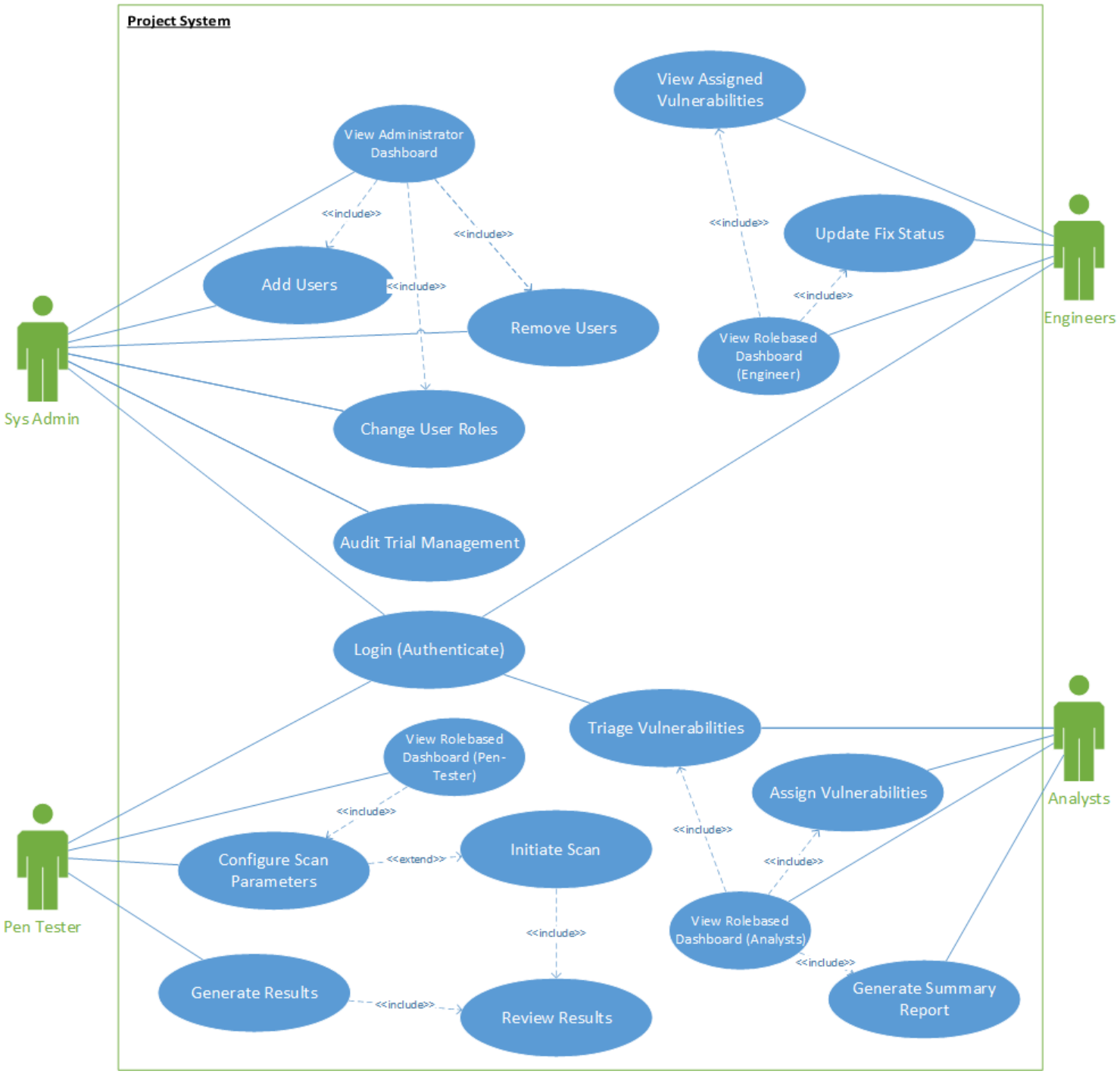

Figure 2: Use Case Diagram

6.2 Detailed Use-Case Descriptions

This section presents detailed Use-Case Descriptions, designed to provide an in-depth understanding of the various interactions and functionalities within the system which have been listed above (6.1). Unlike the high-level use-case abstract, these detailed descriptions delve into the considerations of each interaction, offering a more complete picture of the system's behaviour, (Usability.gov, 2013).

Key terms for reading each use case:

- **Use-Case Name**: The title that identifies the use-case.
- **Description**: A brief summary of what the use-case is supposed to accomplish.
- **Actors**: Individuals that carry out the use-case.
- **Preconditions**: Conditions that must be met before the use-case can start.
- **Postconditions**: Conditions that should be true after the use-case has been carried out.
- **Trigger**: The event that initiates the start of the use-case.
- **Basic Flow**: The ideal sequence of steps that occur when everything goes as planned.
- **Alternate Flow**: Variations in the sequence of steps that occur when there are unusual problems or errors within the basic flow.

  (Usability.gov, 2013).

Detailed Use-Cases

Use Case 01	**Description**
Use-Case Name	Login (Authenticate)
Description	User logs into system to gain access.
Actors	System Admin, Pen Tester, Engineer, Analyst
Preconditions	The user must have an active account.
Postconditions	The specific user will have an authenticated session active.
Trigger	Logging into System
Basic Flow	**Step 1**: User opens the GUI **Step 2**: User inputs their Username and Password **Step 3**: The system checks the entered credentials against the database **Step 4**: If the credentials are valid, the system will grant access and redirects the user to their role specific dashboard.
Alternate Flow	**Step 4(a):** If the credentials are invalid, the system will display an error message indicting that the entered credentials are incorrect

Use Case 02	**Description**
Use-Case Name	View Role Specific Dashboard
Description	Allows the user to view the dashboard associated with their specific.
Actors	System Admin, Analyst, Pen Tester, Engineer.
Preconditions	User is authenticated.
Postconditions	User views their role specific dashboard
Trigger	User logs into the system
Basic Flow	**Step 1:** User Logs into the System. **Step 2:** User is directed to their dashboard
Alternate Flow	**N/A**

Use Case 03	**Description**
Use-Case Name	Add Users
Description	Allows the administrator to add new users to the system/database.
Actors	System Admin
Preconditions	Administrator is authenticated and authorised to add new users.
Postconditions	A new user is added to the system with specified role.
Trigger	Administrator selects the “Add New User” option.
Basic Flow	**Step 1**: Admin logs into the admin dashboard. **Step 2:** System admin navigates to the “User Management” section. **Step 3**: System admin clicks on the “Add User” button. **Step 4:** System admin fills out the user details form. **Step 5**: System admin clicks the “Submit” button. **Step 6**: System validates the information and adds the user to the database. **Step 7**: System displays a confirmation message.
Alternate Flow	**Step 5(a):** If the form has incomplete or invalid data, an error message will be displayed and a prompt for the admin to correct the invalid details.

Use Case 04	**Description**
Use-Case Name	Audit Trail Management
Description	Allows the admin to view the audit trail, which will log all system activities.
Actors	System Admin
Preconditions	Administrator is authenticated and authorised to view the audit trail.
Postconditions	Admin views or exports the audit trail.
Trigger	Admin selects the " View Audit Trail" option.
Basic Flow	**Step 1**: System admin logs into the admin dashboard. **Step 2**: System admin navigates to the Audit Trail. **Step 3**: System admin views the logs. **Step 4**: System admin can export the logs as a CSV or PDF file
Alternate Flow	**N/A**

Use Case 05	**Description**
Use-Case Name	Remove Users
Description	Allows an Admin to remove an existing user from the system/database.
Actors	System Admin
Preconditions	Administrator is authenticated and authorised to add new users.
Postconditions	The selected User is removed from the system/database.
Trigger	Administrator selects the “Remove User” option.
Basic Flow	**Step 1**: Admin logs into the system. **Step 2**: System admin navigates to the "User Management" section. **Step 3**: System admin clicks on the “List Users" button. **Step 4**: System admin selects a user. **Step 5**: System admin clicks the “Remove User” button **Step 6**: System validates the information and adds the user to the database. **Step 7**: System displays a confirmation message.
Alternate Flow	**Step 5(a):** If the action is cancelled, the user will not be removed, and the system returns to the "User Management" section.

Use Case 06	**Description**
Use-Case Name	Configure Scan Parameters
Description	Allows the Pen Tester to configure the parameters for a scan
Actors	System Admin
Preconditions	Pen tester is authenticated to configure the scope of a scan.
Postconditions	Scan parameters are set.
Trigger	Pen Tester starts a scan process. This will be their first requirement pre-initiation.
Basic Flow	**Step 1**: Pen Tester logs into the system.
	Step 2: Pen Tester selects the "Initiate Scan".
	Step 3: Pen Tester sets the scan parameters such as target IP range, scan type etc.
	Step 4: System validates the parameters and sets them for the subsequent scan
Alternate Flow	**Step 4(a):** If the system detects incomplete parameters, it will display an error message and expect a correction.

Use Case 07	**Description**
Use-Case Name	Initiate Scan
Description	Allows the Pen Tester to initiate a vulnerability scan based on the configured scope.
Actors	Pen Tester
Preconditions	Pen Tester is authenticated and scan parameters have been set.
Postconditions	A scan is initiated.
Trigger	Pen Tester selects the imitate scan option and has configured the scan parameters.
Basic Flow	**Step 1**: Pen Tester logs into the system.
	Step 2: Pen Tester selects the "Initiate Scan".
	Step 3: Pen Tester sets the scan parameters such as target IP range, scan type etc.
	Step 4: System validates the parameters and sets them for the subsequent scan
	Step 5: System Displays the scan progress.
Alternate Flow	**Step x**: If any errors are encountered during the scan, it will display an error message and stop the scan.

Use Case 08	**Description**
Use-Case Name	Generate Results
Description	Allows the Pen Tester to generate a report based on the completed scan.
Actors	Pen Tester
Preconditions	A scan has been completed.
Postconditions	A scan report is generated.
Trigger	Pen Tester selects the “Generate Results” option on a selected scan.
Basic Flow	**Step 1**: Pen Tester logs into the system.
	Step 2: Pen Tester selects “Completed Scans” option. **Step 3**: Pen Tester selects a scan. **Step 4**: Pen Tester clicks “Generate Comparative Report” **Step 5**: System will generate a report.
Alternate Flow	**N/A**

Use Case 09	**Description**
Use-Case Name	Review Results
Description	Allows the Pen Tester to review the results of a completed scan.
Actors	Pen Tester
Preconditions	A vulnerability scan has been completed and results generated.
Postconditions	Pen Tester reviews the scan results.
Trigger	Pen Tester selects the "Review Results" button on selected scan results.
Basic Flow	**Step 1**: Pen Tester logs into the system.
	Step 2: Pen Tester selects the "Review Results". **Step 3**: Pen Tester views the scan results, including identified vulnerabilities, their severity, and recommended actions.
Alternate Flow	**N/A**

Use Case 10	**Description**
Use-Case Name	Assign Vulnerabilities
Description	Allows the Analyst to assign identified vulnerabilities to a specific team member for remediation.
Actors	Analyst
Preconditions	Analyst is authenticated and vulnerabilities have been identified
Postconditions	Vulnerabilities are assigned to specific team members
Trigger	Analyst has selected a “reviewed” flagged scan.
Basic Flow	**Step 1**: Analyst logs into system.
	Step 2: Filters scan by “Reviewed”
	Step 3: Selects a specific scan
	Step 4: Selects the Engineer and assigns them the vulnerability for mitigation.
	Step 5: System updates the database and confirmation is displayed
Alternate Flow	**Step 4(a):** If there is an issue with the assignment, an error will be displayed
	Step 4(b): If the Engineer has no capacity, an error will be displayed

Use Case 11	**Description**
Use-Case Name	Generate Summary Report
Description	Allows the Analyst to generate a summary report of identified vulnerabilities and their status flags.
Actors	Analyst
Preconditions	Analyst is authenticated.
Postconditions	A summary report is generated.
Trigger	Analyst selects the "Generate Summary Report" option on a specific set of results.
Basic Flow	**Step 1**: Analyst logs into the system.
	Step 2: Analyst selects a specific set of scan results.
	Step 3: Analyst clicks “Generate Summary” button
	Step 4: System generates a summary report based on the identified vulnerabilities in the scan and their status
Alternate Flow	**N/A**

Use Case 12	Description
Use-Case Name	Triage Vulnerabilities
Description	Allows the Analyst to prioritise identified vulnerabilities based on their severity.
Actors	Analyst
Preconditions	Analyst is authenticated.
Postconditions	Vulnerabilities are prioritised.
Trigger	Analyst selects the "Triage" button on a specific Summary report.
Basic Flow	**Step 1**: Analyst logs into system
	Step 2: Analyst selects a specific Summary Report. **Step 3**: Analyst clicks the “Triage” button. **Step 4**: Analyst prioritises the vulnerabilities based on severity and impact. **Step 5**: System validates the triage and updates the database.
	Sequential steps of an alternative scenario
Alternate Flow	**N/A**

Use Case 13	Description
Use-Case Name	View Assigned Vulnerabilities
Description	Allows the Engineer to view vulnerabilities that have been assigned to them for remediation.
Actors	Engineer
Preconditions	Engineer is authenticated and (Minimum one) vulnerabilities have been assigned.
Postconditions	Engineer views the list of their assigned vulnerabilities.
Trigger	Engineer logs into the system.
Basic Flow	**Step 1**: Engineer logs into system
	Step 2: Engineer navigates to their “Assigned Vulnerabilities”. **Step 3**: System displays a list of vulnerabilities assigned to the Engineer.
Alternate Flow	**Step 2(a):** If no vulnerabilities have been assigned, an error is presented detailing same.

Use Case 14	**Description**
Use-Case Name	Update Fix Status
Description	Allows the Engineer to update the status of a vulnerability that they are working on.
Actors	Engineer
Preconditions	Engineer is authenticated and has assigned vulnerabilities.
Postconditions	The status of the vulnerability is updated.
Trigger	Engineer selects a vulnerability from the list of assigned vulnerabilities.
Basic Flow	**Step 1**: Engineer logs into system **Step 2**: Engineer navigates to their “Assigned Vulnerabilities”. **Step 3**: Engineer selects a vulnerability from the list. **Step 4**: Engineer updates the status of the vulnerability to "In Progress", "Fixed", or "Needs Review". **Step 5**: Engineer clicks the "Update Status" button. **Step 6**: System validates the update and modifies the database. **Step 7:** System displays a confirmation message.
Alternate Flow	**N/A**

7. Wire-Frames

The below diagrams are wireframes that serve as a skeletal blueprint of the systems user interface. These wireframes are an integral part of the UI/UX design process, visually outlining the elements that will appear and shaping the overall interaction design, (Academy, 2021).

Login Dashboard Wireframe

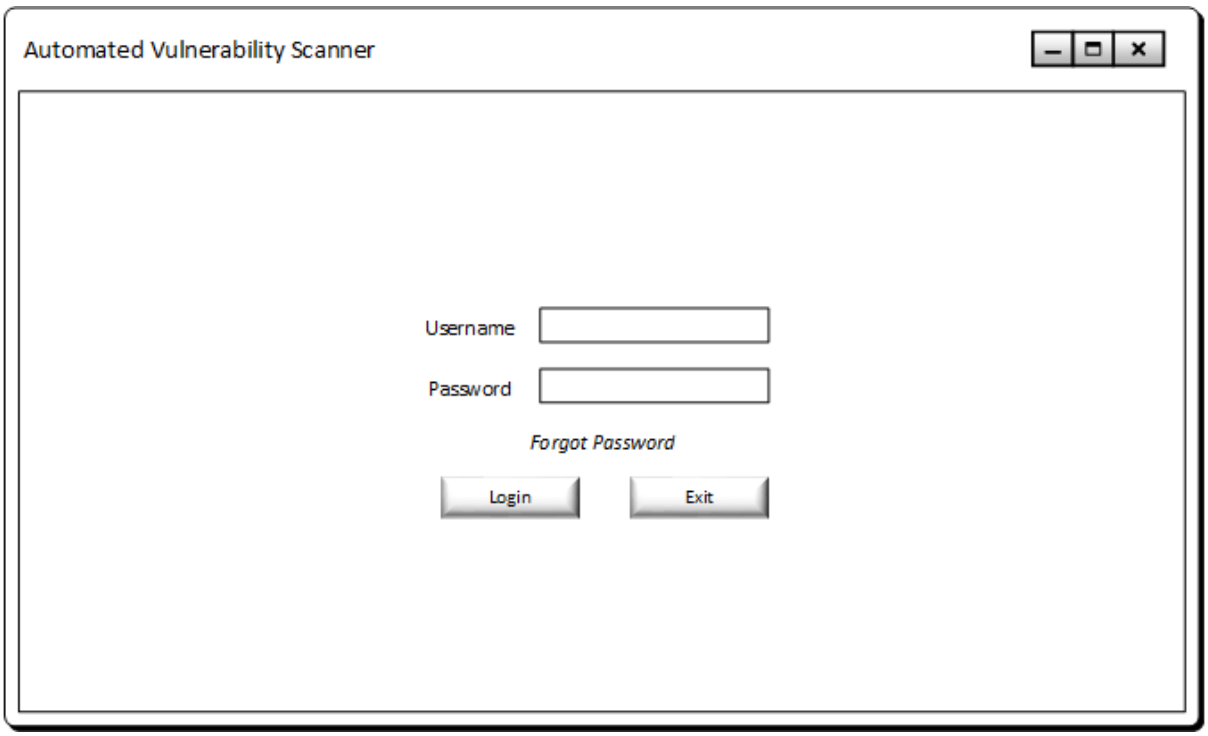

Figure 3: Login Screen of System

System Administrator Dashboard

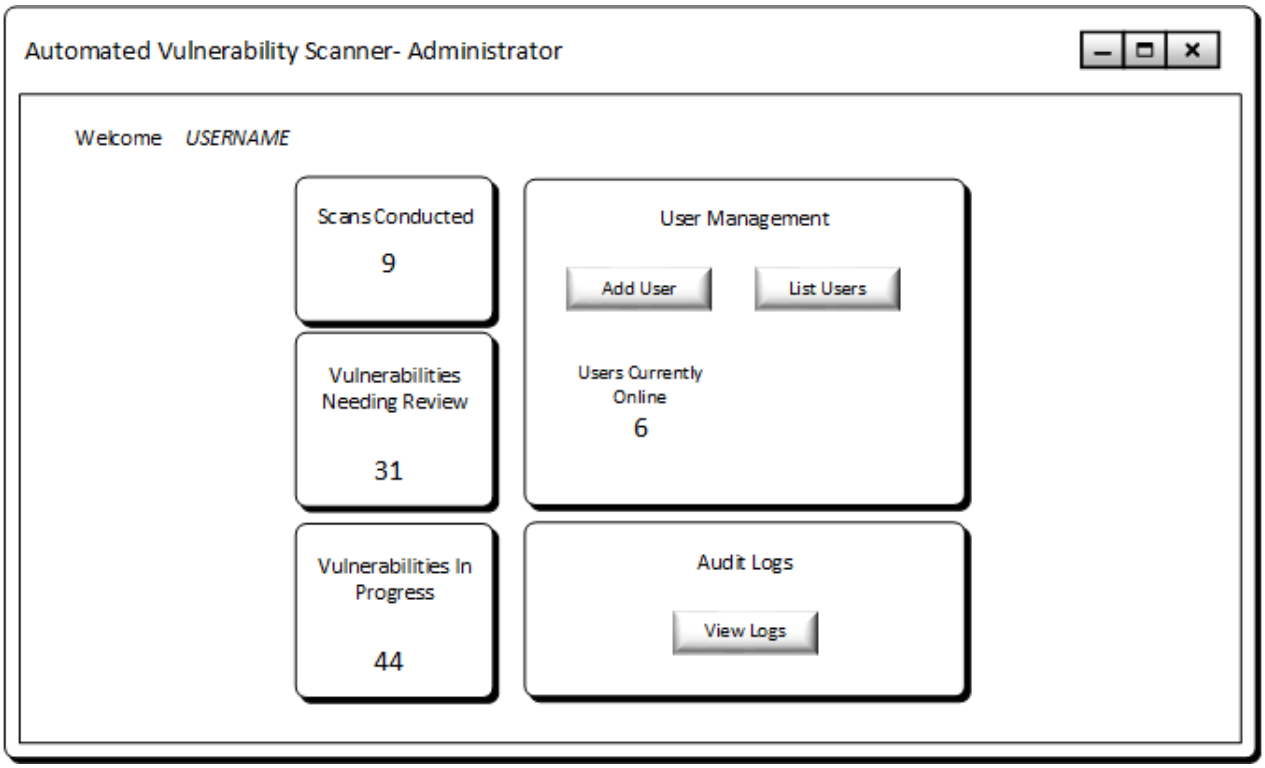

Figure 4: System Administrator Dash

Analyst Dashboard

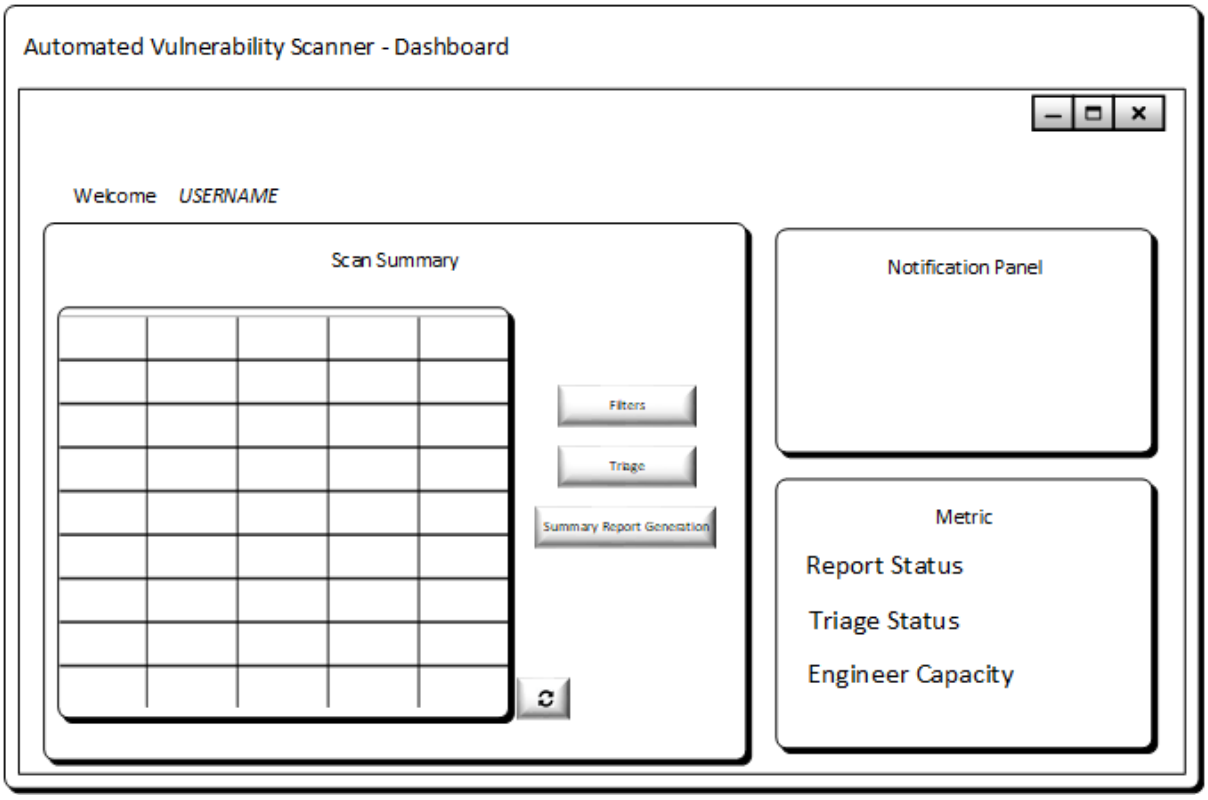

Figure 5: Analyst Dashboard

Engineer Administrator Dashboard

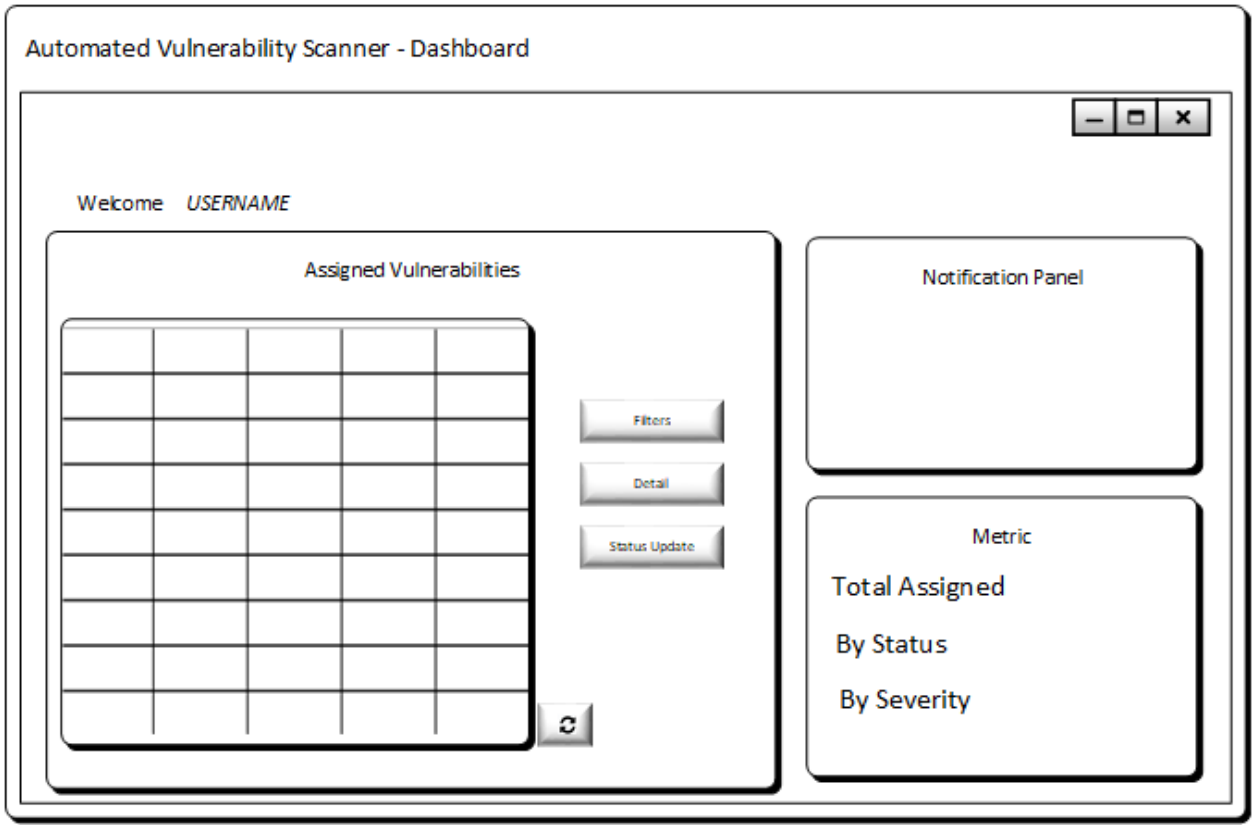

Figure 6: Engineer Dashboard

8. Database Encryption

To ensure data security, AES-256 encryption in Galois/Counter Mode (GCM) has been chosen for the database. AES-256 offers a high level of cryptographic strength that is difficult for threat actors to breach and so safeguards against unauthorised data access (team, 2023), and GCM further enhances this by combining Counter Mode (CTR) with authentication. This makes it faster, more secure, and better optimised for table-driven field operations as well as supporting both authenticated encryption and authenticated decryption, providing a robust security framework, (Kariyawasam, 2021). Pythons' cryptographic library, "cryptography" will be used to handle this process as it offers built-in support for AES-256-GCM, (Cryptography Development Team, 2023).

A separate key database will be maintained to securely store the encryption keys, ensuring that even if unauthorised access to the central database occurs the keys will remain protected. This ensures the confidentiality of data but also its integrity making it an ideal solution for securing the database, (OWASP Cheat Sheet Series Team, n.d.).

The below diagram presents a Data Flow Diagram that outlines the encryption and decryption processes within the system.

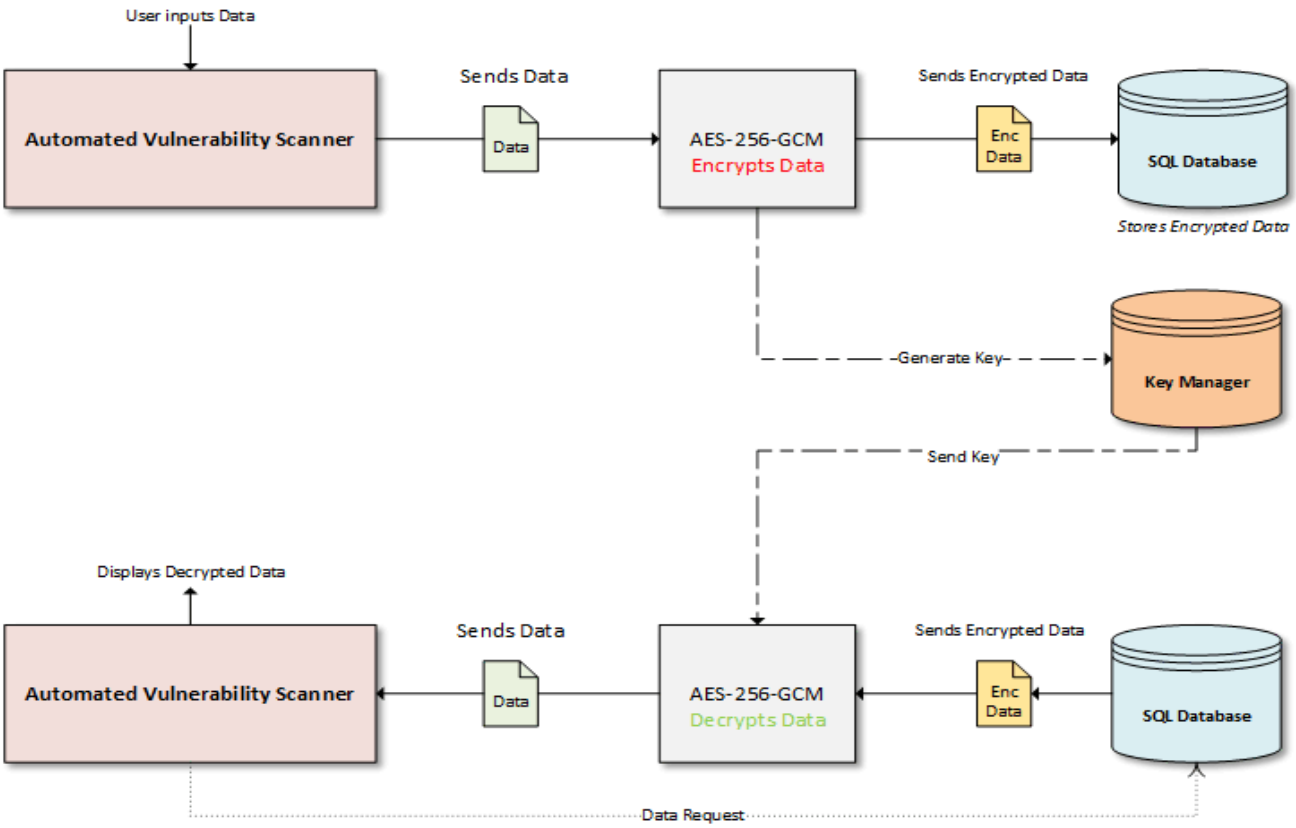

Figure 7: Encryption Data Flow Diagram

9. Project Plan

This section outlines the approach to planning the delivery of this system. To manage the delivery, a phased approach has been proposed. These phases, together with the milestones that underpin them, are outlined below.

9.1 Methodology

9.1.1 Phase 1: Preparation

1. Documentation (Project Specification and Plan, Research Documentation)

The first milestone for the project is the completion of the Project Specification and Plan (this document) and the Research Documentation. The Project Specification will serve as the blueprint for the entire project and detail the overall plan for development. While, the Research Documentation will focus on gathering any information relevant to the project, such as existing solutions and tool comparisons. These two documents will guide the project's direction and ensure there is a clear understanding of the deliverables.

2. Scripting Framework Design and Schema Planning

The design of the Scripting framework will serve as a blue print for how the python scripts will interface with the specific selected tools (see Tools and Technologies). Pseudo code will be created and used as a guide for the development phase. The schema detailing all tables and their relationships will be designed here also which will be used later in order to create the database in the development phase.

3. Initial GUI Design

An initial design phase for the GUI establishing several key elements. The wireframe will be created which will serve as the skeletal outline of the layout and will sketch out placement of elements such as the buttons, text fields and other components. This will aid in development.

9.1.2 Phase 2: Development

1. Database Creation

The docker container will be setup and initialised with an SQL image. Utilising the planned schema, the database will be created and start to serve as the foundational layer for storing data (Scan results, reports etc.).

2. Development of the Unified Vulnerability Assessment and Management System

This involves writing the individual python scripts for controlling each module of the system; which will include:

- *Network Recon Scripts*; Python Scripts for controlling *Nmap*, *OpenVAS*, and *OWASP ZAP*, ensuring they can perform can perform scans of selected networks.
- *Reporting Scritps*; Python Scripts for handling the data output from these scans and formatting this data into a user-friendly report using *ReportLab*.
- *Scheduled Scans*; Managing scheduling repeat scans using Cron.
- *Logical Report Comparison Logic*: Creating the python logic for comparing new scan results with previous ones to identify any changes that may have occurred in-between.

3. GUI Development

The wireframe and pseudo code which was designed in phase one will be used here and translated into working code in order to develop a functional GUI for the system.

4. Comparison Logic Integration

Integration of the comparison logic into the scripting framework and GUI, ensuring that the script can successfully compare new scan results with previous ones and highlight any changes that may have occurred since the initial or previous scan.

5. Scheduling Integration

Integrate Celery into the scripting framework, ensuring that the script can schedule repeat scans at user-defined intervals.

9.1.3 Phase 3: Testing

1. Testing

Thorough testing of the system will be carried out, making sure that all scripts are functioning with the tools and are working correctly and interacting as intended outputting the desired results.

This will include tests with various network types, web application types, and intervals for scheduling.

The GUI will also be tested here to ensure that all components, authentication method and dashboards are functioning as intended and all different roles can conduct their specific use-cases.

9.1.4 Evaluation and Maintenance

1. User Feedback and System Evaluation

Gather user feedback from peers and evaluate the effectiveness of the system this information can then be used to identify any areas of improvement in order to perform maintenance and make any necessary improvements

9.2 Plan Dates and Gant Chart

Outlined below is an overview of the milestones contained within each Phase of the plan, together with proposed start and finish dates.

	Name	Duration	Start	Finish
	Phase I: Project Documentation	50 days	**25/09/23 08:00**	**01/12/23 17:00**
	Project Spec and Plan	25 days	25/09/23 08:00	27/10/23 17:00
	Research	25 days	30/10/23 08:00	01/12/23 17:00
	Phase II: Development	68 days	**04/12/23 08:00**	**06/03/24 17:00**
	Database Creation	1 day	04/12/23 08:00	04/12/23 17:00
	Network Recon Scripts	5 days	05/12/23 08:00	11/12/23 17:00
	Reporting Scritps	5 days	11/12/23 16:00	18/12/23 16:00
	Scheduled Scans	5 days	18/12/23 16:00	25/12/23 16:00
	Report Comparison Logic Development	10 days	26/12/23 08:00	08/01/24 17:00
	Comparison Logic Integration	21 days	09/01/24 08:00	06/02/24 17:00
	GUI Development and Integration	21 days	07/02/24 08:00	06/03/24 17:00
	Phase III: Testing	8 days	**07/03/24 08:00**	**18/03/24 17:00**
	System Testing	8 days	07/03/24 08:00	18/03/24 17:00
	Phase IV: Final Documentation	0 days	**18/03/24 17:00**	**18/03/24 17:00**
	Final Documents and Website	0 days	18/03/24 17:00	18/03/24 17:00

Figure 8: Visual Representation of Plan including Start and Finish Dates

The below chart provides an illustration of the above outlined plan, (the red below illustrates the critical path).

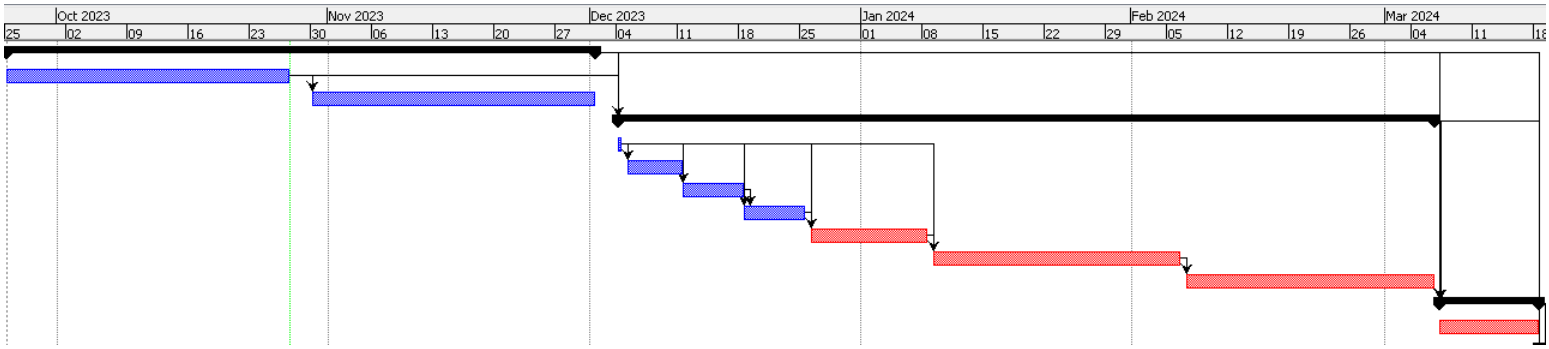

Figure 9: Gant Chart Representation of Plan

10. Metrics using FURPS

"FURPS", which stands for Functionality, Usability, Reliability, Performance and Supportability, is a framework developed by Hewlett-Packard designed for evaluating the quality of software systems, (Dyson, 2019). The model has been widely used to categorise both the functional and non-functional specifications of systems and services, (1000sourcecodes, n.d.).

FURPS will be used as the metric indicator for the system, allowing for a structured approach to measure each of these criteria, thus ensuring that a mechanism for evaluating the degree to which the system meets its intended requirements.

Outlined below are a sample of the expected considerations that will be used to assess whether the system meets its intended requirements.

Functionality

1. **Asset Discovery and Network Scanning**
    - Does the system accurately identify open ports?
    - Is the DNS resolver effectively gathering DNS records?
2. **Vulnerability Scanning**
    - Does the system accurately identify vulnerabilities
    - Is the OWASP ZAP integration working as intended to scan web applications
3. **Data Collection**
    - Does the scanning engine parse raw data effectively?
4. **Report Generation**
    - Are the generated reports detailed?
    - Are the generated reports easy to read and understand?
    - Are the generated reports well formatted
5. **Benchmarking**
    - Does the comparison logic work correctly when comparing new scan results with previous reports?
6. **Scheduling**
    - Is the scheduling function operating?

Usability

1. **User Interface**
    - Is the GUI intuitive?
    - Are the role-based dashboards user-friendly?
2. **Exclusion List**
    - Is it easy for pen testers to customise the scan parameters
3. **Admin Interface**
    - Is the admin interface easy to use for managing users?

Reliability

1. **Data Storage**
    - Is the database handle data correctly and is the data encrypted correctly?
2. **Scheduling**
    - Do the scheduled scans run reliably at the specified intervals?
3. **Benchmarking**
    - Is the comparison logic reliable for analysis?

Performance

1. **Scanning Speed**
    - How quickly does the system complete a network scan?
2. **Report Generation Time**
    - How long does it take to generate an initial report?
    - How long does it take to generate a comparative report?

Supportability

1. **Error Handling**
    - Does the system provide meaningful error messages?
    - Does the system correctly handle errors at all?

2. **Upgradeability**
    - Can modules be added to the system easily?

11. Inspiration

The inspiration for this project came about during my work placement, where I observed that the security tools I was asked to use, such as Nmap, Nessus and Dark web scanner, often all worked in isolation and required manual input that I thought could benefit from increased automation which would in turn create capacity and scope within this process. After evaluating this further, preparing some design considerations, and obtaining permission, I created a script in Python to automate Nmap scans. This allowed me to input IP addresses into a text file for Nmap to read, scan, and export the results into a CSV file which meant I ended up with more working capacity and was freed me up to focus my attention on different aspects of the report holistically cutting down the required time spent on creating these reports. This experience highlighted to me that there was a possible need for a more integrated, automated approach to vulnerability assessment that could enable highly skilled resources to redistribute their time and attention from manual activities to activities where that skillset could be applied in a more effective way.

A platform that resonated with me due to its holistic, innovated approach while on work placement was Stellar's InSoc Open XDR platform, which works by collecting data from existing security tools focused on threat detection and response and presents it on a single dashboard with the making it easier to identify and manage security issues in a more streamlined manner (Stellar-Cyber, n.d.). This inspired me, as I thought I could incorporate the same logic but instead of focusing on threat detection, I could use several vulnerability tools and unify them into one system.

12. References

1. 1000sourcecodes (n.d.) Software engineering-FURPS, Best Online Tutorials | Source codes | Programming Languages. Available at: https://www.1000sourcecodes.com/2012/05/software-engineering-furps.html (Accessed: 26 October 2023).

2. Academy, D. (2021) What is wireframing in UI/UX Design, Medium. Available at: https://medium.com/detaux/what-is-ui-ux-wireframe-designerrs-46dac9c8a153#:~:text=Wireframing%20in%20UI%2FUX%20Design%20is%20one%20of%20the%20most,of%20the%20interaction%20design%20process. (Accessed: 21 October 2023).

3. Adams, C. (2018) Benefits of the graphical user interface, ThoughtCo. Available at: https://www.thoughtco.com/benefits-of-graphical-user-interface-1206357 (Accessed: 27 October 2023).

4. Admins (2023) The challenges of cyber security for smes, Evolution Recruitment Solutions. Available at: https://evolutionjobs.com/exchange/cyber-security-sme/#:~:text=Team%20size%20%E2%80%93%20SMEs%20often%20have,funds%20to%20put%20towards%20it. (Accessed: 18 October 2023).

5. Anglin, S. (n.d.) Why use docker for databases and how: Web with Mizouzie, web with mizouzie (Alt + H). Available at: https://www.mizouzie.dev/articles/why-use-docker-for-databases-and-how/ (Accessed: 26 October 2023).

6. Ansell, M. (2023) Using owasp zap to find web app security vulnerabilities - triad article, Triad. Available at: https://www.triad.co.uk/news/owasp-zap/ (Accessed: 22 October 2023).

7. Aprilliant, A. (2021) Getting started with Cron job in the linux server: A complete tutorial for beginner, Medium. Available at: https://towardsdatascience.com/create-your-first-cronjob-on-the-linux-server-74e2fdc76e78 (Accessed: 22 October 2023).

8. Artykov, D. (2021) Network discovery and security auditing with Nmap, Medium. Available at: https://medium.com/purple-team/network-discovery-and-security-auditing-with-nmap-323432e54ee6 (Accessed: 22 October 2023).

9. Bada, M. and Nurse, J.R.C. (2019) The social and psychological impact of cyberattacks, Emerging Cyber Threats and Cognitive Vulnerabilities. Available at:

https://www.sciencedirect.com/science/article/abs/pii/B9780128162033000046 (Accessed: 27 October 2023).

10. Bajo, A. (2023) Web crawling with python, ScrapingBee. Available at: https://www.scrapingbee.com/blog/crawling-python/ (Accessed: 24 October 2023).

11. Clarke, H. (2023) Benefits of Modular Architecture: Moving from monolithic to Modular, DevOps/SRE Recruitment Experts. Available at: https://www.harrisonclarke.com/blog/benefits-of-modular-architecture-moving-from-monolithic-to-modular (Accessed: 26 October 2023).

12. Cryptography Development Team, (2023). 'Authenticated Encryption with Associated Data (AEAD)', Cryptography Documentation. Available at: https://cryptography.io/en/latest/hazmat/primitives/aead/#aes-gcm (Accessed: 22 October 2023).

13. Das, A. (2021) Top 10 security assessment tools, IndiumSoftware. Available at: https://www.indiumsoftware.com/blog/top-10-security-assessment-tools/ (Accessed: 22 October 2023).

14. Debar, H. (2022) Cybersecurity: High costs for companies, The Conversation. Available at: https://theconversation.com/cybersecurity-high-costs-for-companies-110807 (Accessed: 27 October 2023).

15. Dnspython Contributors(a) (n.d.) *About dnspython*, *dnspython*. Available at: https://www.dnspython.org/about/ (Accessed: 26 October 2023).

16. Dnspython Contributors(b), n.d. dnspython. Read the Docs. Available at: https://dnspython.readthedocs.io/en/latest/ (Accessed: 22 October 2023).

17. Dyson, J. (2019) Conjoining FURPS and Moscow to analyse and prioritise requirements, LinkedIn. Available at: https://www.linkedin.com/pulse/conjoining-furps-moscow-analyse-prioritise-jonathan-dyson (Accessed: 27 October 2023).

18. Fonseca, A.L. (2023) 10 use case diagram examples (and how to create them), 10 Use Case Diagram Examples (and How to Create Them). Available at: https://venngage.com/blog/use-case-diagram-example/ (Accessed: 21 October 2022).

19. Fransen, B. (2023) The imperative of Cybersecurity in a Digital World, EcoMatcher. Available at: https://www.ecomatcher.com/the-imperative-of-cybersecurity-in-a-digital-world/#:~:text=In%20a%20world%20where%20digitization,details%20to%20sensitive%20corporate%20infor

mation. (Accessed: 22 October 2023).

20. IARM (2023) Why is a vulnerability assessment critical for your business? IARM Information Security. Available at: https://www.iarminfo.com/why-is-a-vulnerability-assessment-critical-for-your-business/ (Accessed: 20 October 2023).

21. Ibec (n.d) Sustaining smes, IBEC. Available at: https://www.ibec.ie/influencing-for-business/ibec-campaigns/sustaining-smes/sustaiing-smes (Accessed: 27 October 2023).

22. Indeed Editorial Team (2023) 10 uses of SQL (with definition, benefits and examples). Available at: https://www.indeed.com/career-advice/career-development/sql-uses (Accessed: 26 October 2023).

23. Intelligence, T. (2023) Proactive cybersecurity - what is it, and why you need it, Evolve Security Automation and Orchestration by Threat Intelligence. Available at: https://www.threatintelligence.com/blog/proactive-cybersecurity (Accessed: 22 October 2023).

24. Jaiswal, Ashish and Dwivedi, Aditya. (2021). Python: The Versatile Language - Recent Trends in Programming Languages. 08. 2021. 10.37591/RTPL.

25. Kariyawasam, I. (2021) Selecting the best AES block cipher mode (AES-GCM vs AES-CBC), Medium. Available at: https://isuruka.medium.com/selecting-the-best-aes-block-cipher-mode-aes-gcm-vs-aes-cbc-ee3ebae173c (Accessed: 22 October 2023).

26. Lake Ridge (n.d) 4 reasons small business doesn't invest in cybersecurity, Lake Ridge. Available at: https://www.lakeridge.us/4-reasons-companies-dont-invest-in-cybersecurity (Accessed: 22 October 2023).

27. Lucid chart team (n.d.) What is an entity relationship diagram (ERD)?, Lucid chart. Available at: https://www.lucidchart.com/pages/er-diagrams (Accessed: 22 October 2023).

28. McBee, M. (2022) Trends that underscore the seriousness of the Cybersecurity Skill Gap, Acunetix. Available at: https://www.acunetix.com/blog/web-security-zone/trends-that-underscore-the-seriousness-of-the-cybersecurity-skill-gap/ (Accessed: 18 October 2023).

29. MSFT (n.d) Choose an encryption algorithm - SQL server, SQL Server | Microsoft Learn. Available at: https://learn.microsoft.com/en-us/sql/relational-databases/security/encryption/choose-an-encryption-algorithm?view=sql-server-ver16 (Accessed: 18 October 2023).

30. OWASP Cheat Sheet Series Team (n.d.). Cryptographic Storage Cheat Sheet. OWASP Cheat Sheet Series. Available at: https://cheatsheetseries.owasp.org/cheatsheets/Cryptographic_Storage_Cheat_Sheet.html

(Accessed: 22 October 2023).

31. Pedamkar, P. (2023) Python tkinter: Reasons why do we use Python Tkinter, EDUCBA. Available at: https://www.educba.com/python-tkinter/ (Accessed: 27 October 2023).

32. Python Software Foundation (n.d.) SMTPLIB - SMTP protocol client, Python documentation. Available at: https://docs.python.org/3/library/smtplib.html (Accessed: 26 October 2023).

33. Radečić, D. (2021) How to schedule python scripts with cron - the only guide you'll ever need, Medium. Available at: https://towardsdatascience.com/how-to-schedule-python-scripts-with-cron-the-only-guide-youll-ever-need-deea2df63b4e (Accessed: 26 October 2023).

34. ReportLab (n.d.) Chapter 1: Introduction, Chapter 1: Introduction - ReportLab Docs. Available at: https://docs.reportlab.com/rml/userguide/Chapter_1_Introduction/ (Accessed: 26 October 2023).

35. Saxena, A. (2023) 10 best vulnerability scanning tools you must know, Sprinto. Available at: https://sprinto.com/blog/vulnerability-scanning-tools/#:~:text=Many%20organizations%20use%20a%20combination,of%20different%20features%20and%20options. (Accessed: 17 October 2023).

36. Smith, K. (2022) How to meet the challenges behind vulnerability scans and reports, SC Media. Available at: https://www.scmagazine.com/perspective/how-to-meet-the-challenges-behind-vulnerability-scans-and-reports (Accessed: 20 October 2023).

37. Solar Winds Team (n.d.) What is SQL database? - it glossary, SolarWinds. Available at: https://www.solarwinds.com/resources/it-glossary/sql-database (Accessed: 26 October 2023).

38. Stellar-Cyber (n.d.) Stellar Cyber / Open XDR: Cybersecurity for msps and mssps, inSOC. Available at: https://in-soc.com/stellar-cyber-open-xdr/ (Accessed: 27 October 2023).

39. Surve, S. (2023) Top 10 python libraries for ethical hacking: Enhancing security assessments and vulnerability testing - information array, InformationArray.com. Available at: https://blog.informationarray.com/top-10-python-libraries-for-ethical-hacking-enhancing-security-assessments-and-vulnerability-testing/ (Accessed: 26 October 2023).

40. Tahalani, S. (2020) Owasp Zap-crawl the web app, Medium. Available at: https://hackerman007.medium.com/owasp-zap-crawl-the-web-app-d1500b7e73e2 (Accessed: 26 October 2023).

41. team, K. (2023) Unlocking the Power of AES-256 Encryption: A Comprehensive Guide, Kiteworks. Available at: https://www.kiteworks.com/secure-file-sharing/unlocking-the-power-of-aes-256-encryption-a-comprehensive-guide/#:~:text=The%20key%20strength%20of%20AES,unauthorized%20access%20to%20the%20data. (Accessed: 22 October 2023).

42. Tutorials Point (n.d.) Python - sending email, Online Tutorials, Courses, and eBooks Library. Available at: https://www.tutorialspoint.com/python/python_sending_email.htm (Accessed: 22 October 2023).

43. Usability.gov (2013) Usability.gov. Available at: https://www.usability.gov/how-to-and-tools/methods/use-cases.html#:~:text=A%20use%20case%20is%20a,when%20that%20goal%20is%20fulfilled. (Accessed: 21 October 2023).

44. Vocal Tech (n.d) AES GCM Encryption Algorithms, VOCAL Technologies. Available at: https://vocal.com/cryptography/gcm-and-gmac-authenticated-encryption-algorithms/ (Accessed: 22 October 2023).

45. Vulnerability scanner: What is it and how does it work? (n.d) Snyk. Available at: https://snyk.io/learn/vulnerability-scanner/ (Accessed: 21 October 2023).

46. Zurro, P. (n.d.). Top 13 Vulnerability Scanners for Cybersecurity Professionals | Core Security Blog. Core Security. Available at: https://www.coresecurity.com/blog/top-14-vulnerability-scanners-cybersecurity-professionals. Accessed: 01/10/2023