# Unified Vulnerability Scanning Engine and Management System

Research Document

Andrew Gilbey C00263656

Supervisor: Richard Butler

## Table of Contents

## Abstract

This document details the proposed methodology of the creation of an Integrated Vulnerability Scanning System, which aims to streamline the vulnerability scanning process by combing existing open-source tools into one solution. The key research has focused on evaluating existing vulnerability scanners, how they work, documenting manual and automated scanning. A comparative analysis of several programming languages has been conducted in order to determine the most appropriate language for the development process. Operating Systems have also been compared in order to determine the best development environment for the project.

Several Vulnerability Scanning tools have been researched for implementation considering including, Nmap, Masscan, OpenVAS and OWASP ZAP, with a view to determining which would align the best to the project specifications. In summary, this research is aimed at developing the set of blueprints for the project system and how its goals can be met.

# 1. Introduction

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

This report provides an overview of various research domains which have been explored for the development of the proposed system. It covers several key areas which are as follows -

- **Understanding Vulnerability Scanners and The Process**- An examination of how vulnerability scanner's function and the process they follow.
- **Comparative Analysis of Programming Languages for System Development** – An analysis of different programming languages in order to determine how suitable they would be for building the system.
- **Optimal Operating System for Development Environment** – An examination of Windows and Linux systems to identify which is the most optimal for a development platform for the system.
- **Open-Source Scanning tools Evaluation** – A comparative analysis on different open-source vulnerability scanning tools that could be integrated into the system.
- **Database Hosting Platform and Encryption Methods** – Investigating which platform would be optimal to host the database and which AES encryption method should be used in order to encrypt and decrypt data going into the database.
- **Data Parsing** – Outlining the methods that will be used for parsing data within the system and technique used for generating the main report.
- **Decision on the Final Technology Stack** – Finalising the technology stack which will be used for the development of the project.
- **Definition of User Roles on the System** – Outlining the various different user types that will be using the system and functionalities that they may require.
- **Risk Assessment and Project Scope** – Identifying possible risks that will be associated with the development of the system.

This research document concludes with a summary of the outcomes of the above sections.

# 2. Integrated Vulnerability Scanning Systems

## 2.1 Understanding Vulnerability Scanners and The Process

Vulnerability scanners are a type of specialised software used by cybersecurity teams, typically deployed by businesses to assess their cybersecurity posture and ensure a limited number of vulnerabilities within their network, systems, and applications. There is an array of tools available in the vulnerability scanning landscape, all contributing to the establishment of a vulnerability management programme (Constantin, 2020).

Vulnerability scans are categorised into two types: internal and external. External scans focus on evaluating vulnerabilities of servers and applications that are open to the internet, gauging the risk of external cyber-attacks. Internal scans, on the other hand, pinpoint weaknesses within the local network that could be exploited by hackers accessing the internal network, (Timonera, 2023).

The scanning process involves defining the scope, which means identifying and defining the range of systems or networks to be covered to ensure that no critical asset is missed or irrelevant systems are mistakenly included (Ot, 2023). Once defined, deciding which systems to scan is vital. Businesses use a mix of devices and cloud services, each requiring a different scanning approach: exposure-based, sensitivity-based, and coverage-based, (Andrew, n.d). The exposure-based strategy focuses on the most attack-prone assets, such as internet-facing websites and applications, which are vital for maintaining the business's reputation. The coverage-based method evaluates the potential impact and range of threats posed by vulnerabilities, weighing the risks of different assets, like the difference in threat level between a breach in an employee's laptop versus the main server. In a sensitivity-based approach, attention is centred on safeguarding assets containing critical data, such as stored customer information, (Data Dome Learning Center, 2021).

Once the scope and parameters are set and the scan is run, the next step involves reporting and analysis. Vulnerability scanners are equipped with customisable reporting features, which focus on the most critical vulnerabilities found. Rigorous analysis is key to uncovering deeper vulnerabilities, especially true after a prolonged period without scanning. It's important to categorise each finding by its risk and severity score, ensuring that vulnerabilities can be addressed quickly and properly, thus minimising the risk of them being overshadowed by more severe issues, (Ot, 2023).

To summarise, the vulnerability scanning process begins with an extensive assessment and discovery phase where all devices and systems on the network, including servers, workstations, routers, and firewalls, will be identified. This is followed by defining the scope and parameters of the scan, deciding which assets are to be included and whether it should be conducted internally or externally. Once the specific scanning tool is selected, the scanning process commences, checking each asset within the scope for vulnerabilities by sending tests and analysing the responses. After the scan is completed, the report is generated and analysed, leading to the final stage of mitigating and remediating the identified vulnerabilities through measures such as applying security patches or reconfiguring systems (Laurente-Ticong, 2023).

### 2.1.1 Manual Approach vs Automated Approach to Vulnerability Scanning

Vulnerability scanning can be conducted either manually or automated. Manually scanning is a process where cybersecurity team members will closely inspect systems in order to offer a precise identification of vulnerabilities, this method however, is both resource extensive and time consuming, (Red Node, 2023).

An automated scanner is designed to autonomously identify security vulnerabilities in a target 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). Because of the nature of manual scanning, it would not be feasible for this project and instead automated scanning would be utilised. By harnessing the capabilities of automated scanners, this would allow for a rapid identification of vulnerabilities and to ensure a consistent detection of vulnerabilities, (Malik, 2023).

### 2.1.2 Challenges with Vulnerabilities Scanners

Vulnerability scanners are almost an essential tool for cybersecurity teams- however, they do come with their own list of challenges which can impact their effectiveness, efficiency and reliability. Some of these challenges would be issues relating to accuracy, such as flagging false positives/negatives; the complexity and requirement for a more specialised user for expertise, (Ali, 2023); and integration or compatibility issues with the currently in place infrastructure, (Violino, 2022). Each of these challenges can and will complicate the process of effectively identifying security vulnerabilities and therefore in turn affect the mitigating of potential threats.

False Flagging should be highlighted due to the inherent risk associated with them, this issue takes the appearance of two forms: Type I errors which are false positives, where a non-issue vulnerability is highlighted which leads to the possibility of unnecessary remediation and wasted time. More serious are the Type II errors, false negatives. These are vulnerabilities that will remain undetected and lead to a false level of assurance among security teams. To combat these issues, scanner results should be verified and a wider range of tools should be used in a multi-layered approach, (Consultant IT, 2020).

The complexity inherent in scanners goes beyond their basic function and moves onto the specialised knowledge which is necessary for their effective application, (Ali, 2023). As some of the tools are advanced in nature, it requires users to have a deeper level of technical understanding in order to efficacy utilise the scanners. Team members responsible for the scanning processes must be skilled in not just running the scanners but also correctly interpreting outcomes, (Neumetric, 2023). This need for a specialised expertise can present a problem, particularly for smaller businesses that may lack resources to have a dedicated cyber security team with cybersecurity professionals, (Microsoft 365 Team, 2019). This would suggest that these smaller businesses may benefit from a more streamline approach and supports the development of an integrated system whereby all the tools are in one, centralised location.

Integration challenges within an already existing IT infrastructure can present a hurdle, again this is especially true for smaller businesses.

There are a multitude of security tool available, each with their own distinct capability and often operate in isolated silos, working individually, which leads to a cumbersome integration process. This situation can overwhelm even senior IT professionals as integrating newer tools into existing systems without the proper communication and insurance of compatibility can obscure the threat landscape. There are on-going efforts within the cybersecurity community, for example; the Open Cybersecurity Alliance, to address these issues by developing common data models and open standards to improve interoperability, (Violino, 2022) – this also supports the concept of this project which would aid in the integration challenge and also ensure a clearer cybersecurity strategy with an emphasis on the target demographic of small to middle sized enterprises (SMEs).

### 2.1.3 Compliance within the Vulnerability Process

Compliance with regulatory standards such as the General Data Protection Regulation (GDPR) and various ISO standards is specifically important, especially when it comes to vulnerability scanning, (Breachlock Research, 2023). The GDPR, which was introduced in 2018, emphasis a strict control of the private data of individuals within EU and requires businesses to implement spiralised measures to ensure data security, although to does not explicitly mention or cover penetration testing or vulnerability assessments, article 32 of the GDPR implicit necessitates regulator testing and evaluating the effectiveness of security measures, (Intersoft Consulting, 2016). ISO standards provide a framework for establishing, implementing and continually improvising a security management system and underlines the need for a structured approach to managing protecting information assets. Obedience to these regulations means to have a strict line of measures for scanning and would require further staff training, access controls and audit trails to ensure data integrity, (Irwin, 2023).

This pushes forward that this project must implement a secure security measure in order to align with, especially so with GDPR and to include some level of role-based access control and audit trails.

# 3. <u>Programming Languages</u>

## 3.1 <u>Python</u>

Python is a high-level, interpreted programming language which is renowned for its readability due to its straightforward syntax, which also support programming paradigms such as object orientated and functional programming. This simplicity makes it accessible and an ideal choice for developers. Python has a multitude of libraries, such as Scrapy for packet manipulation, PyCrypto for cryptography and PtyhonDNS for DNS reconnaissance, which could prove to be invaluable for the project system. These libraries are also heavily supported by a large and active community.

Python does have limitations; because it is interpreted, it may not perform as well as compiled languages such as C++ in resource hungry tasks and this may make it potentially less suitable for high-performance requirements. Python also has a Global Interpreter Lock (GIL) which can act as a bottleneck in multi-threaded applications and so limit its effectiveness in scenarios that require any concurrent processing. While Python is widely used and heavily supported, these weaknesses are important to consider.

## 3.2 Java

Java is a class-based and object-orientated programming language, which follows the principle “write once, run anywhere” capability which allows any Java program to be compiled on a single platform and executed across any platform featuring the java virtual machine (JVM), which means this “platform independence” gives Java a very high-degree of portability, (Payne, 2023).

Java utilises the Java Standard Library (JSL), which offers a large range of pre-built code modules for tasks such as I/O operations, data structures, networking and error handling which can support in efficient development of an application, (Kumar, A., n.d.) and this could be ideal for development of the project system.

Javas’s platform independence does come at the price of performance, Java programs run slower than natively compiled languages (C/C++) and it has a longwinded syntax which can result in a longer development time and would require a greater effort to maintain, (Winter, 2023). Java also restricts low-level access to system resources in order to increase security which makes it less suitable for system-level applications that may require hardware manipulation, (Payne, 2023), although this would have no direct consequence for the project system. It is also worth noting that since 2019, oracle requires a paid commercial license for business, commercial or production for the use of Java 9, which introduces a cost factor for using certain enterprises features of Java, (Neville, 2023), as the project system will be designed for leveraging open-source tools, this commercial element to Java, even if not impacting the project can be regarded as a conflict.

## 3.3 Ruby

Ruby is an open-source programming language, which has a high emphasis on simplicity. It was designed with the intention of making programming both fun and easy and so features a syntax with easy readability which tries to mirror natural language, (ACG Technical Editors Team, n.d.). It also supports multiple programming models, including but not exclusive to, object-orientated, procedural and functional programming, making Ruby versatile for any type of software development, (Stabile, C.U., 2023).

Rubys primary feature, RubyGems, is a package management system that allows developers to import various add-ons such as libraries and code packages (known as gems) which would allow developers to easily add functionality to their projects without “reinventing the wheel”. This therefore can significantly reduce time and effort required to implement intricate functionality and speed up the development process.

Ruby does however, have performance limitations, especially in high-efficiently projects, and its application speed can be considered to be slower when compared to other programming languages and so in turn could cause scalability issues, this however, in smaller projects, may not be considered a problem. The language can also be thought of as a “niche” language and so lacks a large community of support when compared to languages such as a Python. As whole Ruby would probably be more suited for web-applications rather than standalone applications.

## 3.4 C++

C++ although considered a high-level language, has low level functionality especially when c compared to other modern-day languages and is used widely to create high-performance applications. C++ provides extensive control over system resources and memory allocation due to its low-level capabilities and so allows direct manipulation of memory through pointers and enable developers to manage memory allocation and deallocation clearly. This level of control is critical for optimising performance which is why C++ is used in such systems as Operating systems, Game engines and real-time systems, (Rassokhin, D., 2020).

Having low-level functionality, C++ therefore comes with certain complexities. Features such as pointers and lambdas can be daunting and if mis-used can lead to system crashes. The strict syntax is not as flexible as in other languages and this can extend the coding time and errors and as It has such a strict standard for errors any simple programming errors can lead to a horde of errors which can over-complicate the debugging process, (Yörü, 2023).

While C++ offers great control for system resources, it's complexity and difficult error analysis may impact the development efficiency for the system project.

## 3.5 Programming Language Comparative Conclusion

In summary, Python stands out for its versatility, the abundance of compatible libraries for Python, such as PythonDNS and PyCrypto, and over an all familiarity with the language. This means utilising python would be especially advantageous for the project and it's large and supportive community would be a significant asset which could provide invaluable assistance during the development phase.

When directly compared to other the languages, Pythons familiarity give it a clear advantage over C++, which despite its power, poses complexity challenges. Ruby, while user-friendly, falls short due to its niche appeal and performance limitations. Java, would be considered the second strongest in contention, due to familiarity; however, it lacks the extensive library support and simplicity of Python, which may slow down the development process. Therefore, it can be concluded, Python is the most suitable choice of programming language for the project going forward.

# 4. Operating System Development Environment

## 4.1 Windows Vs Linux

In the development of the system project, a decision must be made for the operating system. This choice will be influenced by the systems performance, compatibility with vulnerability scanning tools and the overall development experience. This section will evaluate two operating systems; Windows and Linux, and gauge their strengths and weaknesses in relation to the project requirement.

### 4.1.1 Performance and Reliability

Linux has the reputation of having a notable speed increase and better performance reliability than windows, even under strain, and this can be attributed to its streamlined design and optimised resource management. Unlike Windows, which often runs numerous background processes that consumes large amount of RAM, Linux maintains a lean operational profile. Linux also has a well organised file system, where data is stored in closely located chunks, facilitates faster read-write operations in contrast to windows less structured file storage approach, (Software Testing Help Team, 2023). Recent testing from July, 2023 has shown that several Linux distributions out performed Windows 11 in CPU, GPU and memory benchmarks, (Morgan, 2023).

This performance would be critical for any time-intensive tasks and for use in a development environment and so would suit the needs of the project far more than Windows.

### 4.1.2 Development Environment

Linux stands out as the superior choice for a development environment for the project. In terms of compatibility with essential scanning tools like OpenVAS, Nmap, Burp Suite and Nessus and the ease of automation of these tools in Linux adds to its appeal.

Windows however, despite supporting several tools such as Nmap and Burp Suite, it falls behind achieving the same level of integration as Linux. One example is OpenVAS, which does not offer direct computability with Windows and would require setting up a Linux VM, introducing an extra layer of complexity which may disrupt the workflow in a Windows-Based environment and cause significant slow-down.

Given these considerations, selecting Linux for the project would offer a more beneficial environment for development than Windows. It's far superior capabilities for tool integration and its system performance would solidify Linux as the optimal choice going forward.

## 4.2 Linux Distribution Comparative – Ubuntu Vs Kali

To specify using Linux would be considered too generic, as a Linux Distribution (Distro) is essentially a version of Linux which is packaged with additional software in order to specific to certain requirements. These vary in their focus and features and offer a wide range of options for different use cases.

For this research document two Distribution are considered – Ubuntu and Kali

Ubuntu is considered an excellent platform for programming tasks and is easy to use. It features a rolling release cycle, which ensures that developers have access to the latest software updates at all time, which is important for staying up to date with Integrated Development Environments (IDE)s and tools. It also supports various desktop environments such as GNOME and KDE which provides a high level of customisation, which means it is possible to tailor the work space for specific needs and preferences. It offers full computability with the vulnerability scanners in contention but will require additional installation steps.

Kali is tailored specifically for security testing and features the majority of vulnerability scanning tools out of the box. KALI also has a high level of customisation similar to ubuntu which again would the development environment can be tailored to specific preferences

# 5. Open-Source Scanning Tools

There are several Open-source scanning tools that are freely available (open-source) and for the most part developed by a community of security experts, (Kime, C, 2023). The range of these open-source scanning tools encompasses a magnitude of different types which would be tailored to a specific need. Network scanners such as Nmap, provide network reconnaissance, reporting on open ports and services across a network, (Nmap Team, n.d.) and other scanners such as OpenVAS that got a step further, and assess systems completely and provide detailed reports, (Greenbone AG, n.d.). Web application scanners such as OWASP ZAP are specifically developed to find security flaws in web applications such as cross-site scripting, SQL injection and other web application-based vulnerabilities, (OWASP, n.d.). This arsenal of open-source tools can all be utilised by businesses in order to tighten up their security posture within the current digital landscape, (Kime, C, 2023).

## 5.1 Key Open-Source Tools

Each of the following tools can be utilised in different aspects of vulnerability scanning and are widely recognised by the cybersecurity community for their contribution into maintaining security posture for businesses and even for individual home networks.

### 5.1.1 Nmap

Nmap (Network Mapper) is a widely used tool in the IT industry and has a reputation as a versatile network discovery and security auditing tool which would make it a valuable asset for the project. Nmap excels in efficiently mapping network environment and can identify what hosts are available, the services of which they are running, the operating systems they are being run on and the types of packet filers or firewalls in use, (Bogert,J., 2022).

Integrating Nmap into the project system would potentially augment the systems capability to map out networks as its first layer of scan and as Nmap is written in C and LUA it has a good compatibility with the programming language Python and so integration into Python can be done quite easily, (Study Tonight Team, n.d.).

Nmap has the functionality of being able to export its results in XML format, and the structure nature of XML allow for simple parsing and transformation into different formats, (Nunez, 2022), most noticeably CSV or PDF files. By utilising Python libraries such as Etree, Report lab or Pandas, it would be possible to parse the XML results into CSV or PDF, (Mike, 2012), in order to include them into a main report as well as extract any information in order to include them into a database.

Nmap does not have a built in API for integration, but outputs can be easily parsed due to its XML or CSV export capability, (Schultz & Perciaccante, 2017, pp. 100–101).

### 5.1.2 Masscan

Masscan is a Network scanner similar to Nmap and is popular tool within the cybersecurity community due it's exceptional speed and ability to conduct large-scale network scans, this added capability positions it as another potential asset in the project as an alternative to Nmap. Masscan is much the same to Nmap in that it can scan open ports and detect services that are currently in use across network, however Masscan has the capacity to scan the internet in under 6 minutes when given the right conditions and this speed sets apart from not only Nmap but other network scans in the same category, (Biniyaz, J.K., 2022). Despite its impressive speed, Nmap is still regarded as the most versatile tool in the field, (Miessler, n.d.), and offers a more detailed network analysis.

The output of Masscan can be exported in JSON and XML format which offers flexibility in handling the scan results, (Innokrea Team, 2023), python libraries such as Etree can also be utilised to parse the data and transform them into CSV formats or PDF.

Similar to Nmap, Masscan does not have a built in API however, output can be exported in XML and JSON files, for integration into other systems.

### 5.1.3 Open VAS

OpenVAS is a versatile vulnerability scanning tool which leverages a database of over 120,000 vulnerability signatures, which enables the identification of a vast range of vulnerabilities which include misconfigurations, web application vulnerabilities and deviations away from standard security standards, (Federal Office for Information Security (BSI), 2023). The strength in OpenVAS lies in its diverse testing methods, it can run server audits, compliance checks, web application testing as well as network vulnerability scanning ensuring a thorough evaluation of potential weaknesses in a scanned network, (Greenbone AG, n.d.).

OpenVAS's adaptability supplies to the needs of different organisations and its active database is a is regularly updated to keep pace with any emerging vulnerabilities, (Federal Office for Information Security (BSI), 2023). It is worth noting recent updates to OpenVAS have introduced new features designed to augment the scanners effectiveness, (Greenbone AG (a), n.d.).

OpenVAS however, does have limitations. It covers less Common Vulnerabilities and Exposures (CVE)s when compared to some competitors such as (commercially available) Nessus, and as such can lead to missing specific vulnerabilities, (Keary, 2022). It also can be considered a "complex" tool and so additional technical knowledge may be required for its effective use which can create a challenge when interpreting results, (Guardian, 2023).

While OpenVAS is a capable tool, it may not match the coverage offered by commercially available vulnerability scanners, but can be considered a multi-layered tool and does well in detecting known vulnerabilities.

OpenVAS supports an API, the OpenVASManagement Protocol (OMP) which allows for the configuration of scans as well as data extraction, (Verma, 2021), which would be useful for integrating OpenVAS into the project system and automating its vulnerability scanning capabilities.

### 5.1.4 Burp Suite

Burp suite Is a tool developed by PortSwigger Ltd, and contains of a set of tools with the aim of testing web applications for vulnerabilities, effectible combining both penetration testing utilities and a vulnerability scanner within a single tool. Burp Suite is renowned for its ability it intercepts traffic between server and browser which allows for an in-depth analysis and re-rounded testing approach for web applications, (PortSwigger (a),2023).

Some included tools are a Proxy, Repeater, Decoder, Sequencer and Comparer (these are in its free community edition version) which all allow for a detailed assessment for the target web app by intercepting and manipulating traffic in order to compare responses, (PortSwigger (b), n.d.).

Burp Suite does support an API but only in the professional version of the scanner, (PortSwigger(c), 2023), which is a commercial product. This is a challenge for the project, which aims to primarily use open-source tools and so rely on a commercial product would contradict the open-source objective.

### 5.1.5 OWASP ZAP

OWASP ZAP is another open-source tool that is primarily used for testing the security posture of web applications, (Homola, O., 2023). It easy to implement, install and is an accessible tool for those new to application testing, it is also compatible with the big three Operating systems (Linux, Windows and MAC OS), (Savaliya, K.  2023; Moradov, 2022). It acts as a man-in-the middle proxy and functions between the browser and the targeted application, which allows it to intercept and analyse any exchanged information, identifying security vulnerabilities.

ZAP take a dual scanning approach to its functionality, broken into active and passive scanning. Active scanning uses knock attack patterns to detect vulnerabilities while Passive scanning non-intrusively monitors any HTTP requests/responses to ensure a thorough detection process, (Moradov, 2022). ZAP also has a spider tool, which works to crawl through all pages within a web page, essentially mapping the web application, (Hiremath, O. 2020).

Despite the strengths, ZAP does have limitations. The tools interface is outdated and it can be clunky and requires customisation in order to achieve a better user-experience, (Homola, 2023), this is not strictly applicable to the project however, as automated scripts will be utilised – not the interface direct.  The automated experience with ZAP is also limited in terms of detecting the more complex vulnerabilities and again would require extensive customisation for a better analysis which may result in a higher rate of false negatives when compared to the more advanced tools, (Prashant, 2023).

Inclusion of ZAP into the project would be considered beneficial for the web application scanning aspect, as it’s scanning abilities and open-source nature would make it align to the open-source goal of the project and its range of tools would be ideal for the project’s web application scanning requirements. The spider crawler could also be a benefit for use with an exclusion list in the system as it would be able to 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).

## 5.2 Tool Comparative Analysis and Selection

For networking scanning, both Nmap and Masscan are efficient solutions, neither require an API and both would suit the requirements of the project. However, Masscan, while has superior speed, provides a less comprehensive network analysis compared to Nmap. The general familiarity with Nmap and ease of integration, makes Nmap the ideal choice for the project going forward.

OpenVAS is an ideal choice for fulfilling the vulnerability scanning actual of the project, its database and variety of testing methods allows it to achieve the projects requirements and the support an easy-to-use API, which supports scan configuration and data extraction enhances this appeal. This aligns well with the projects need as a whole and OpenVAS therefore should be used going forward.

For application testing, despite Burp Suite's impressive testing capabilities, its API support being limited to a commercial product is a significant roadblock, especially considering the projects inclination toward Open-source tools. OWASP ZAP however, despite some limitations in regards to automation, aligns far more closet to the projects open-source goal and can fulfil the requirement for application scanning.

To conclude, the selection for the project going forward will be as follows;

- **Nmap** to conduct networking mapping and scanning.
- **OpenVAS** to conduct the vulnerability scanning aspect.
- **OWASP ZAP** to conduct the web application scanning.

## 6. Database Solution

The effectiveness of the database solution is crucial, not only for retaining data but as the key component for the report comparison logic. To effectively manage this, the database must reliable be able to capture all outcomes of security scans. Another requirement for the database is to facilitate user-based vulnerability management and it must be able to enable the assignment of specific vulnerably to designated team members, focusing on targeted remediation.

As part of this research element of the project, it is imperative to conduct a comparative analysis of the different technologies that could impact the system and meet requirements. In this context, two critical comparisons; XAMPP vs Docker for the development environment setup and SQL vs MongoDB for the database management approach will follow.

### 6.1 XAMPP & PhpMyAdmin vs. Docker: Development Environment

XAMPP is a popular and open-source web server solution and is one of the most widely used cross-platform web servers. It consists of Apache HTTP server, interpreter and MariaDB, (Javatpoint, n.d.), which employs PhpMyAdmin to manage databases. PhpMyAdmin is an effective tool for managing MYSQL databases and does so through a user-friendly web interface. Its main strength lies in its simplicity and enables users to handle database tasks effortlessly, without any in-depth technical expertise. Other features also include the ability to backup and restore databases, conduct queries and database searches, all of which can streamline the database management for developers, (Kozon, 2023). It is important to note that for the specific requirements a web host is not necessary, the primary need is for a reliable database host that can support efficient management capabilities.

Docker, however, stands out as the most broadly favoured platform for the development of applications in containers. These containers create an isolated space containing both the application and all its required software packages, (Walker, 2023).

Docker when used with MySQL is not without its challenges, notably in runtime resource usage. Databases are designed to be stateful and require persistent storage, this conflicts with the stateless nature of Docker containers. Extending the lifespan of database containers to maintain statefulness can lead to resource inefficiencies, as databases consume a significant amount of CPU and RAM, (Giro, 2020), and this prolonged usage could present a potential hurdle in the future.

In conclusion, Docker is an overall superior choice for hosing the database, particularly highlighting its versatility in creating isolated environments. XAMPP, which primarily focuses on providing a web server solution with an integrated database management tool, Docker offers a more flexible and scalable approach. Despite its challenge in managing stateful applications (Databases), Dockers ability to manage resources effectively makes it a more compelling choice.

## 6.2 Database Encryption Techniques

In order to stay onboard with regulations such as General Data Protection Regulation (GDPR), it is key to encrypt any sensitive data that will be transferred into the databases. Encryption transforms data into a secure, coded unreadable format, that makes it inaccessible to unauthorised users. The focus here will be on comparing the Advanced Encryption Standard (AES) in order to determine what would be a suitable mode for achieving requirements.

### 6.2.1 AES-256 (Counter Mode)

The advanced encryption standard, is a protocol for encrypting and decrypting data and supports key lengths of both 128 and 256 bits, (Sung, Kim, and Shin, 2018). In its Counter Mode (CTR), which is one of the five operational modes of AES, a user-defined initial counter is incremented for each 16-byte data block until overflow, which ensures uniqueness in both the counter and key combination for each block, (Microchip Lightning Support, 2019). CTR mode ability to encrypt/decrypt data blocks independently and in parallel makes it a secure and efficient choice for encrypting databases (Educative Answers, 2023). However, it should be noted that CTR mode does not provide an authentication and integrity mechanism so it must be paired with one, (Housley and Tschofenig, 2023).

### 6.2.2 AES-256 Galois/Counter mode

AES-256Galois/Counter mode (AES-GCM) is built around the AES block cipher and is a preferred choice over the CTR mode alone. It works by combing the AES CTR mode for encryption with Galois Message Authentication code (GMAC) for authentication (McGrew and Igoe, 2015). Each AES-GCM instance features a block counter, which increment with each AES call to produce output blocks and resets with each new request. GCM can also server as a stand-alone message authentication code, which is useful for incrementally authenticating large dynamic datasets and therefore would be ideal for database encryption, (McGrew & Viega, n.d.). Pythons' cryptographic library also supports AES-256-GCM, allowing it to be implemented in the project seamlessly, (Cryptography Development Team, 2023).

# 7. Data Parsing

## 7.1 Data Extraction

### 7.1.1 Nmap – Data Extraction

Nmap has two primary output formats; XML and plain text. It has the functionality to save its results directly into an XML file, which can then be parsed into other file formats or directly stored into a database. The XML format is the most convenient for this project as its structure is machine-readable, making it easy to interpret using Python libraries, (Schultz & Perciaccante, 2017, pp. 100–101).

### 7.1.2 OpenVAS – Data Extraction

Data from OpenVAS can be extracted through several methods; through the Greenbone Security Assistant, which is essentially the OpenVAS GUI, which provides a button that can be clicked to extract necessary scan data, for this project however this method would not be applicable. The second method, is using the Greenbone Vulnerability Manager Command Line Interface (GVM-CLI) which enables a direct interaction with the vulnerability manager through the CLI and so data extraction can be achieved with automated scripts, (Greenbone AG (b), n.d.). OpenVAS also has its API, OMP, which can also be used to extract data, (Verma, 2021), which is another viable option, and would allow VAS to be seamlessly added to a dashboard which would fit requirements for the project.

Data can be extracted in several formats which include XML, HTML, CSV and PDF. Given the specific requirements, XML or CVS would be the most logical choices due to structure, enabling them to be easily parsed, (Greenbone AG (c), n.d.).

### 7.1.3 OWASP ZAP – Data Extraction

Similarly, data extraction with OWASP ZAP can be achieved through either the REST API that allows interacting with ZAP scans, through the use of the ZAP GUI, for a more manual extraction process and the ZAP Command line Interface (CLI). The API can be used to pull out scan results and would be considered particularly useful for integrating ZAP into dashboards, (REST Api, n.d.).

ZAP supports several file formats, however there are three that are applicable, these are; XML. JSON and Markdown (MD), again XML would be the most appropriate here but JSON files could also be utilised for similar reasons – they are machine-readable and the data can be easily parsed.

## 7.2 Parsing Strategy

This is the ideal method which will be used to interpret and process the data when converting it from one format to another. This will involve takin the results of each individual scan and convert them into a unified format before integrating them into the main report. This will involve the follow steps:

Scans must first be run using their respective tools, each tool (Nmap, OpenVAS, OWASP ZAP) contain their own feature to export the results into an XML format. Once generated, the data will need to be parsed and the relevant information extracted. In Python this can be done using libraries – some XML parsing libraries are documented below:

**Xml.etree.ElementTree** – This is a built-in library that provides efficient API for parsing XML data. It is part of the standard library and does no require any extra effort in order to setup, (Python Docs, 2023).

**lxml** – This is a third-party library that provides an extensive set of tools for processing XMLs – especially XMLs of larger size, (Behnel, 2022).

**BeautifulSoup** – BeautifulSoup although typically used in parsing HTML files, can be utilised in XML parsing, (Eshiett, 2023).

A script would then be required to read in each XML and pull the necessary data which can then be added piece-by-piece to the main report, which will be in the PDF file format.

## 8. Decided Technology Stack

This section documents the decided technology stack based on the research up until this point;

**Kali Linux** – Kali was selected as the primary operation system, Kalis preloaded with a majority of tools, and operates on Linux, making it faster and more reliable than Windows.

**Python –** Python will be used for its familiarity and vast library support, which means it can easily interact efficiently with the scanning tools in the project stack.

**Docker –** Docker will host the database, again for familiarity with the system and its containerisation approach allows it to be portable, thus is an ideal choice for the database host.

**Identified Python Libraries**;

- **xml.etree.ElementTree** - Will be used for the XML parsing of scan result data exports.
- **PythonDNS -** Will be used in order to scan DNS records of a given domain which will be added in the recon section of the main report.
- **ReportLab**- Is utilised for generating the main report as a pdf after data has been parsed and compiled into one singular report.

**Scanning tools**;

- **Nmap** to conduct networking mapping and scanning due the general familiarity with the tool and its ease of integration.
- **OpenVAS** to conduct the vulnerability scanning aspect, as it has a suitable vulnerability database and has variety of testing methods. It also supports an easy-to-use API.
- **OWASP ZAP** to conduct the web application scanning, it aligns well with projects open-source goal, is easy to use and has functions such as the spider tool which will add useful functionality for the project.

## 9. User Roles and Accessibility

The system will have to be designed to have a different functionality dependant on the type of user that is accessing it at a given time. This is important  from a security and compliance stand point, as differentiating access based on the different working roles will ensure that users only have access to the functions necessary for their role and therefore reduces the risk of sensitive data being accessed inappropriately.

The roles that will be included in the system are as follows;

**System Administrator (Sys Admin)-** System admins will have the highest level of access in order to oversee the system. Their will have the responsibility of managing the system, adding and removing users and adjusting user roles where necessary.

**Penetration Tester (Pen Tester)** – Pen testers are responsible for configuring the scan parameters, initiating the scans and generating reports, which are then forwarded to Analysts for triage.

**Analyst** – Analysts are responsible for the analysis of the reports, find and prioritising the critical vulnerably found and then delegating tasks based on urgency and the engineer's capacity.

**Engineer** – Engineers will be focused on resolving the identified vulnerabilities and applying the necessaire patches or changes to mitigate risks and ensure the vulnerabilities are dealt with.

As each role defined will have different responsibilities, they will of course require the system to have different functions and when logged in must be presented with a dashboard that is tailored to their specific requirements. 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.

# 10. Project Risks

## 10.1 Security, Legal and Compliance Risks

Legal concerns are at the forefront of considered risks for the project. Under the Criminal Justice (Offences Relating to Information Systems) Act 2017, conducting unauthorised scans on any network is considered illegal, (Criminal Justice (Offences Relating to Information Systems) Act 2017, s. 2). This means, before any scan from the system is conducted, explicit permission from the system owners must be sought after. This includes a clearly defined scope of consent which must be agreed too and documented before any scanning can commence.

Regulatory and compliance risks are also a potential risk, explicitly in relation to the GDPR, as compliance with GDPR is essential when handling any personal data that may be collected during a scanning process, (Intersoft Consulting, 2016). This involves securing any necessary consent for data processing and ensuring encryption implemented to protect the data stored, (Intersoft Consulting, n.d.).

## 10.2 Technical Risks

The tech stack, theoretically should be fully compatible for integration with the selected tools and enable efficient parsing of data and its consolidation into a single main pdf report for analysis, however, the process of integrating these tools to work together may present unforeseen challenges such as version compatibility, data format inconsistences or difficulties in automating the workflow from data extraction to report generation.

## 10.3 In-Operation Risks

When the project is in its operational phase, there is a general risk associated with Human-error. This concerns mistakes such as incorrect scan configuration which even minor and easily done can result in major problems, especially in regards to security and compliance.

## 11. Conclusion

This research document has provided insight into the various research domains explored to support the determination of the features, characteristics and development approach of the most effective project system.

Several key areas have been analysed to support this research, ranging from the function of vulnerability scanners, programming languages and operating systems; to assessing vulnerability scanning tools; database hosting platforms and data encryption methods.

These insights have been examined to enable the definition of key elements of the project system build, including the data parsing strategy, final technology stack and the definition of user roles.

An assessment of key risks associated with the development of the project system have been identified and outlined for consideration as part of the next phases of the project system build.

## Acknowledgments

I would like to extend a thank you to my supervisor, Richard Butler, for his guidance and support throughout this research paper.

## References

1. Admin, S. (2016) The challenges of scaling up web vulnerability scanning, Software Testing Class. Available at: https://www.softwaretestingclass.com/the-challenges-of-scaling-up-web-vulnerability-scanning/ (Accessed: 19 November 2023).

2. Ali, N. (2023) 7 limitations of vulnerability scanners, 7 limitations of Vulnerability Scanners. Available at: https://beaglesecurity.com/blog/article/limitations-of-vulnerability-scanners.html (Accessed: 19 November 2023).

3. Andrew, D. (n.d.) The Ultimate Guide to vulnerability scanning, RSS. Available at: https://www.intruder.io/guides/the-ultimate-guide-to-vulnerability-scanning (Accessed: 18 November 2023).

4. Behnel, S. (2022) Benchmarks and speed, lxml - XML and HTML with Python. Available at: https://lxml.de/3.2/performance.html (Accessed: 10 December 2023).

5. Bogert, J. (2022) Nmap basics: What is nmap &amp; how is it used? Linux Security. Available at: https://linuxsecurity.com/features/nmap-basics-what-is-nmap-how-is-it-used (Accessed: 19 November 2023).

6. Breachlock Research (2023) Penetration testing and vulnerability scanning for GDPR, BreachLock. Available at: https://www.breachlock.com/resources/blog/penetration-testing-and-vulnerability-scanning-for-gdpr/ (Accessed: 19 November 2023).

7. Constantin, L. (2020) What are vulnerability scanners and how do they work? CSO Online. Available at: https://www.csoonline.com/article/569221/what-are-vulnerability-scanners-and-how-do-they-work.html (Accessed: 18 November 2023).

8. Consultant, I.G. (2020) Vulnerability scans and false positives, IT Governance UK Blog. Available at: https://www.itgovernance.co.uk/blog/vulnerability-scans-false-positives-the-importance-of-sanitising-input (Accessed: 19 November 2023).

9. Criminal Justice (Offences Relating to Information Systems) Act 2017, s. 2. Irish Statute Book. Available at: https://www.irishstatutebook.ie/eli/2017/act/11/section/2/enacted/en/html (Accessed: 10/12/2023).

10. 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).

11. Data Dome Learning Center (2021) What is vulnerability scanning? what are is different uses? - datadome. Available at: https://datadome.co/learning-center/what-is-vulnerability-scanning/ (Accessed: 18 November 2023).

12. Educative Answers (2023) Educative answers - trusted answers to developer questions, Educative. Available at: https://www.educative.io/answers/what-is-ctr (Accessed: 06 December 2023).

13. Eshiett, J. (2023) Parsing XML with beautifulsoup in python, Stack Abuse. Available at: https://stackabuse.com/parsing-xml-with-beautifulsoup-in-python/ (Accessed: 10 December 2023).

14. Federal Office for Information Security (BSI). (2023). Open Vulnerability Assessment System. Available at: from https://www.bsi.bund.de/EN/Themen/Unternehmen-und-Organisationen/Informationen-und-Empfehlungen/Freie-Software/Tools/OpenVAS/OpenVAS_node.html (Accessed: 20 November 2023).

15. Greenbone AG (a) (n.d.) Greenbone openvas, OpenVAS. Available at: https://openvas.org/#:~:text=Greenbone%20OpenVAS,any%20type%20of%20vulnerability%20test (Accessed: 19 November 2023).

16. Greenbone AG (b) (n.d.) TechDoc portal. Available at: https://docs.greenbone.net/API/GMP/gmp-21.4.html (Accessed: 09 December 2023).

17. Greenbone AG (c) (n.d.) 11 reports and vulnerability management¶, 11 Reports and Vulnerability Management - Greenbone Enterprise Appliance 21.04.26 documentation. Available at: https://docs.greenbone.net/GSM-Manual/gos-21.04/en/reports.html (Accessed: 09 December 2023).

18. Guardian (2023) OpenVAS: Way to robust vulnerability assessment, CyberAstral. Available at: https://cyberastral.com/cyberworld/cybersecurity/tools/openvas/#:~:text=OpenVAS%20can%20be%20a%20complex,Setting%20Up%20Installing%20OpenVAS (Accessed: 28 November 2023).

19. Hiremath, O. (2020) The owasp zap hud, News and Notes from the Makers of Nexus. Available at: https://blog.sonatype.com/the-owasp-zap-hud (Accessed: 28 November 2023).

20. Homola, I. (2023) OWASP ZAP: 8 core features (pros &amp; cons), Codiga. Available at: https://www.codiga.io/blog/owasp-zap/ (Accessed: 28 November 2023).

21. Homola, I. (2023) OWASP ZAP: 8 core features (pros &amp; cons), Codiga. Available at: https://www.codiga.io/blog/owasp-zap/ (Accessed: 10 December 2023).

22. Housley, R. and Tschofenig, H. (2023) 'CBOR Object Signing and Encryption (COSE): AES-CTR and AES-CBC', RFC 9459, Internet Engineering Task Force (IETF), September 2023, [Standards Track]. Available at: https://www.rfc-editor.org/rfc/rfc9459.html (Accessed: 05/12/2023).

23. Innokrea Team (2023) Masscan (part 2), Applications and software for companies, enterprises, Gdansk. Available at: https://www.innokrea.com/blog/advanced-masscan-options/ (Accessed: 20 November 2023).

24. Intersoft Consulting (2016) Art. 32 GDPR – security of processing, General Data Protection Regulation (GDPR). Available at: https://gdpr-info.eu/art-32-gdpr/ (Accessed: 19 November 2023).

25. Intersoft Consulting (n.d.) Encryption, General Data Protection Regulation (GDPR). Available at: https://gdpr-info.eu/issues/encryption/ (Accessed: 10 December 2023).

26. Irwin, L. (2023) GDPR Article 32: Your guide to the requirements, IT Governance UK Blog. Available at: https://www.itgovernance.co.uk/blog/gdpr-article-32-your-guide-to-the-requirements (Accessed: 19 November 2023).

27. Javatpoint (n.d.) Xampp tutorial - javatpoint, www.javatpoint.com. Available at: https://www.javatpoint.com/xampp (Accessed: 02 December 2023).

28. 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).

29. Keary, T. (2022) Nessus vs openvas: Which is better? A head-to-head comparison, Comparitech. Available at: https://www.comparitech.com/net-admin/nessus-vs-openvas/#:~:text=OpenVAS%20Product%20Highlights%20OpenVAS%20is,OpenVAS%20Scanner%20into%20a%20module (Accessed: 28 November 2023).

30. Kime, C. (2023) 10 best open-source vulnerability scanners for 2023, eSecurity Planet. Available at: https://www.esecurityplanet.com/applications/open-source-vulnerability-scanners/ (Accessed: 19 November 2023).

31. Kozon, T. (2023) Using phpmyadmin for MySQL Database Management - pros and cons, Using PhpMyAdmin for MySQL Database Management - Pros and Cons. Available at: https://boringowl.io/en/blog/the-pros-and-cons-of-

using-phpmyadmin-in-your-php-development (Accessed: 02 December 2023).

32. Kumar, A. (no date) Introduction to java - pw skills - PW skills: Blog, Learn Coding Anywhere Anytime -PW Skills Blog. Available at: https://pwskills.com/blog/introduction-to-java/ (Accessed: 28 November 2023).

33. Laurente-Ticong, L. (2023) What is network vulnerability scanning? complete guide, Enterprise Networking Planet. Available at: https://www.enterprisenetworkingplanet.com/security/network-vulnerability-scanning/ (Accessed: 18 November 2023).

34. Malik, K. (2023) A complete guide to automated vulnerability scanning, Astra Security Blog. Available at: https://www.getastra.com/blog/security-audit/automated-vulnerability-scanning/ (Accessed: 19 November 2023).

35. McGrew, D. and Igoe, K. (2015) 'AES-GCM Authenticated Encryption in the Secure Real-time Transport Protocol (SRTP)', RFC 7714, Internet Engineering Task Force (IETF), December 2015. Available at: https://datatracker.ietf.org/doc/html/rfc7714 (Accessed: Date).

36. McGrew, D.A. and Viega, J. (n.d.) The Galois/counter mode of operation (GCM) - identity digital, CSRC NIST. Available at: https://csrc.nist.rip/groups/ST/toolkit/BCM/documents/proposedmodes/gcm/gcm-spec.pdf (Accessed: 06 December 2023).

37. Microchip Lightning Support. (2019). SAM4E AES Peripheral - AES256 CTR mode. 000007453 Available at: URL (Accessed: Date).

38. Microsoft 365 Team (2019) What small businesses need to know about cybersecurity, Microsoft. Available at: https://www.microsoft.com/en-us/microsoft-365/business-insights-ideas/resources/cybersecurity-small-business#:~:text=,time%20availability (Accessed: 19 November 2023).

39. Miessler, D. (n.d.) Masscan examples: From installation to everyday use, Unsupervised Learning. Available at: https://danielmiessler.com/p/masscan/ (Accessed: 20 November 2023).

40. Mike (2012) Parsing XML and creating a PDF invoice with python, Mouse Vs Python. Available at: https://www.blog.pythonlibrary.org/2012/07/18/parsing-xml-and-creating-a-pdf-invoice-with-python/ (Accessed: 19 November 2023).

41. Moradov, O. (2022) OWASP ZAP: 8 key features and how to get started, Bright Security. Available at: https://brightsec.com/blog/owasp-zap/ (Accessed: 28 November 2023).

42. Neumetric (2023) Breaking down the cost of vulnerability assessments, Neumetric. Available at: https://www.neumetric.com/cost-of-vulnerability-assessments/ (Accessed: 19 November 2023).

43. Neville, M. (2023) Pros and cons of java: Key advantages and disadvantages - softjourn, Softjourn Inc. Available at: https://softjourn.com/insights/pros-and-cons-of-java-development (Accessed: 28 November 2023).

44. Nmap Team (n.d.) Nmap. Available at: https://nmap.org/#:~:text=Nmap%3A%20Discover%20your%20network,monitoring%20host%20or%20service%20uptime (Accessed: 19 November 2023).

45. Nunez, J.V. (2022) How to enhance nmap with python, freeCodeCamp.org. Available at: https://www.freecodecamp.org/news/enhance-nmap-with-python/ (Accessed: 19 November 2023).

46. Ot, A. (2023) How to do a vulnerability scan effectively in 6 steps: ESF, Enterprise Storage Forum. Available at: https://www.enterprisestorageforum.com/software/how-to-do-a-vulnerability-scan/ (Accessed: 18 November 2023).

47. OWASP (n.d.) Vulnerability scanning tools, Vulnerability Scanning Tools | OWASP Foundation. Available at: https://owasp.org/www-community/Vulnerability_Scanning_Tools#:~:text=Web%20Application%20Vulnerability%20Scanners%20are,as%20Dynamic%20Application%20Security (Accessed: 19 November 2023).

48. Payne, R. (2023) Advantages and disadvantages of Java, Developer.com. Available at: https://www.developer.com/java/advantages-and-disadvantages-of-java/#:~:text=,a%20high%20degree%20of%20portability (Accessed: 28 November 2023).

49. PortSwigger (a) (2023) Intercepting HTTP traffic with BURP proxy, PortSwigger. Available at: https://portswigger.net/burp/documentation/desktop/getting-started/intercepting-http-traffic#:~:text=In%20this%20tutorial%2C%20you%27ll%20use,when%20you%20perform%20different%20actions (Accessed: 28 November 2023).

50. PortSwigger (b) (n.d.) BURP suite - application security testing software, PortSwigger. Available at: https://portswigger.net/burp (Accessed: 28 November 2023).

51. PortSwigger(c) (2023) Pricing - BURP suite enterprise edition, PortSwigger. Available at: https://portswigger.net/burp/enterprise/pricing (Accessed: 10 December 2023).

52. Prashant (2023) A comprehensive comparison of OWASP ZAP and BURP suite vulnerability assessment tools - part 2 - cyber security blogs - valency networks -, Cyber Security Blog. Available at: https://www.valencynetworks.com/blogs/a-comprehensive-comparison-of-owasp-zap-and-burp-suite-vulnerability-assessment-tools-part-2/ (Accessed: 10 December 2023).

53. Python Docs (2023) Xml.etree.ElementTree - The Elementtree XML API, Python documentation. Available at: https://docs.python.org/3/library/xml.etree.elementtree.html (Accessed: 10 December 2023).

54. Rassokhin, D., (2020). The C++ programming language in cheminformatics and computational chemistry. Journal of Cheminformatics, [online] 12(1), p.10. Available at: https://doi.org/10.1186/s13321-020-0415-y (Accessed: 28 November 2023).

55. Red Node (2023) Automated VS manual vulnerability scanning, RedNode. Available at: https://rednode.com/insights/automated-vs-manual-vulnerability-scanning-which-one-is-right-for-your-business/ (Accessed: 19 November 2023).

56. REST Api (n.d) Rest api. Available at: https://docs.zapbi.com/en/connect-to-rest-api.html (Accessed: 09 December 2023).

57. Savaliya, K. (2023) How to install owasp zap on windows and linux, Techofide. Available at: https://techofide.com/blogs/how-to-install-owasp-zap-on-windows-and-linux/ (Accessed: 28 November 2023).

58. Schultz, C.P. and Perciaccante, B. (2017) ‘The nmap output formats’, in Kali Linux Cookbook: Effective penetration testing solutions. Birmingham, UK: Packt Publishing Ltd., pp. 100–101.

59. Stabile, C.U. (2023) A brief history of ruby, A Brief History of Ruby. Available at: https://auth0.com/blog/a-brief-history-of-ruby/ (Accessed: 28 November 2023).

60. Study Tonight Team (n.d.) Using the NMAP port scanner with python, Studytonight.com. Available at: https://www.studytonight.com/network-programming-in-python/integrating-port-scanner-with-nmap (Accessed: 19 November 2023).

61. Sung, B.-Y., Kim, K.-B. and Shin, K.-W. (2018) 'An AES-GCM authenticated encryption crypto-core for IoT security', 2018 International Conference on Electronics, Information, and Communication (ICEIC), Honolulu, HI, USA. doi:

10.23919/ELINFOCOM.2018.8330586.

62. Timonera, K. (2023) External vs internal vulnerability scans: Difference explained, eSecurity Planet. Available at: https://www.esecurityplanet.com/networks/external-vs-internal-vulnerability-scan/ (Accessed: 18 November 2023).

63. Verma, A. (2021) Security assessment : Openvas, GeeksforGeeks. Available at: https://www.geeksforgeeks.org/security-assessment-openvas/ (Accessed: 10 December 2023).

64. Violino, B. (2022) 7 top challenges of Security Tool Integration, CSO Online. Available at: https://www.csoonline.com/article/572023/7-top-challenges-of-security-tool-integration.html (Accessed: 19 November 2023).

65. 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).

66. Walker, J. (2023) How to use Docker for your mysql database, Earthly Blog. Available at: https://earthly.dev/blog/docker-mysql/#:~:text=Running%20MySQL%20inside%20a%20Docker,you%20the%20benefit%20of%20consistency. (Accessed: 02 December 2023).

67. Winter, R. (2023) An intro to java for back-end programming, HubSpot Blog. Available at: https://blog.hubspot.com/website/javascript-for-backend (Accessed: 28 November 2023).

68. Yörü , Y. (2023) The Pros and cons of the C++ programming language, Embarcadero RAD Studio, Delphi, &amp; C++Builder Blogs. Available at: https://blogs.embarcadero.com/the-pros-and-cons-of-the-c-programming-language/ (Accessed: 28 November 2023).