PUSL3190 Computing Individual Project

Interim Report

The Basic Security Test Application

Supervisor: Dr. Pabudi Abeyrathne

Name: Pasidu A Liyanage

Plymouth Index Number: 10899322

Degree Program: Bsc(Hons) Computer Security

Table of Contents

Chapter 01 – Introduction

1.1 Introduction

The Basic Security Testing Application solves a fundamental requirement for IT auditing through its reliable and efficient user-friendly risk assessment platform. Security has developed into an urgent priority because of cyber threats which highlights the necessity for accurate audit solutions. The IT auditing market confronts multiple difficulties because their procedures demand excessively long, and error-prone manual testing combined with complicated Nmap and OpenVAS tools which require steep learning curves and manual report generation is inefficient. These problems lead to lengthy audits and unfinished assessments that raise business exposure to risks and increase the chance of cyber incidents together with regulatory violations and reputational damage.

Through automation The Basic Security Testing Application completes essential risk scans which decreases operational costs and expenses and enhances the precision of audit outcomes. Security tests execute in parallel from within the application through its simultaneous running of port scans combined with vulnerability checks and firewall status checks. The system's ability to run tests simultaneously shortens evaluation durations for completing system evaluations particularly when auditing extensive complex network systems.

The platform produces detailed reports automatically which reflect advanced professional standards. The automated system brings simplicity to documentation tasks by eliminating human data compilation mistakes as well as creating standardized reports based on client needs to enhance clarity and improve decision efficiency. Auditors and clients benefit from the generated reports that categorize detected vulnerabilities according to risk levels while offering remediation solutions for resolving serious security issues.

Agile development methodologies enable the project to integrate Python programming as its backend foundation. The structured approach enables ongoing refinement through continuous testing with feedback loops to keep auditors and industries up to date. The Basic Security Testing application serves to enhance corporate cybersecurity practices through better risk assessment efficiency while improving accuracy alongside usability to address current gaps.

1.2 Problem Definition

Security audits are essential to maintaining the integrity of IT systems. However, traditional auditing techniques involve manual processes that are time-consuming and prone to human error. In this role, auditors must look for vulnerabilities in a variety of systems, including networks, software, databases, and hardware components. The lack of an efficient, automated tool significantly impacts the accuracy and speed of security assessments.

Performing risk assessments stands among IT auditors' primary responsibilities because these assessments may require thorough scans of extensive network and system systems. The evaluation follows several sequential steps that need different security testing skills including penetration testing along with network analysis and system configuration checks. System vulnerabilities require immediate attention because untested systems are directly exposed to possible threats. The market now requires a solution which allows IT auditors to conduct their examinations more swiftly without compromising precision or depth of assessment.

IT auditors encounter significant challenges due to reporting problems that occur during hands-on testing and report compilation activities. Auditors need to perform distinct independent assessments throughout the system before they generate their assessment report by collecting findings from each test. The availability of Nessus and OpenVAS tools has drawbacks since they demand complex configuration before achieving successful operation and their usage requires mastery of steep learning curves. A solution which simplifies testing operations and reporting tasks along with reducing complex procedures must be developed because of essential requirements.

The lack of an effective solution for vulnerability scanning and audit report generation can have major consequences for companies and their IT audit teams. Hackers or malicious individuals are more likely to exploit vulnerabilities. Furthermore, limitations can cause auditors to overlook critical security issues. As a result, it is essential to deploy a solution that allows IT auditors to conduct thorough, timely assessments of system vulnerabilities, while limiting oversight risk and ensuring that companies maintain strong cybersecurity defences.

1.3 Project Objectives

Develop and introduce an automated security testing application for IT auditors:
By automating the procedure, the Basic Security Test Application enables auditors to complete audits in a fraction of the time while maintaining accuracy.
Automation will ensure that all critical sections of the system are properly checked for flaws, lowering the possibility of human mistake and detecting vulnerabilities as soon as feasible.

Improve the efficiency and accuracy of vulnerability scanning:
Another key goal of the project is to enable IT auditors to execute various security tests simultaneously. The tool's concurrent testing feature will increase audit efficiency and assist IT auditors in meeting tight deadlines.

Generate detailed, automated security reports:
The goal is to ensure that the application generates detailed, automatic reports following each audit. These reports will summarize the vulnerabilities discovered, prioritize them by severity, and make actionable suggestions for remedy. By automating this procedure, the technology will save auditors time while also ensuring that clients receive clear, professional reports following each audit.

Chapter 02 - System Analysis

2.1 Facts Gathering Techniques

The Basic Security Testing application provides multiple fact-gathering methods which let IT auditors detect and inspect vulnerabilities found in networks and software systems as well as databases while examining system configurations. The system performs automated audit work through the combination of different security scanning tools and reporting tools. The security project relies on the following main fact-gathering approaches:

1. Vulnerability Scanning Techniques
Multiple systematic scanning methods allow the application to detect existing security weaknesses during its operation.
System port scanning establishes an overview of open, closed and filtered ports in order to locate unauthorized access points.

How it works:

- Version Checks:
  The application checks software versions against a database where it identifies existing CVEs related to those versions.
- Network Scanning:
  The tool displays which hosts are active with their services along network paths for security vulnerability analysis.
- Configuration Audit:
  The evaluation tool detects security weaknesses through assessment of system-level port status and default user passwords and incorrect firewall setups.
- Security Libraries:
  Network service and web application scanning can be performed through security tools including Nessus and OpenVAS.

Collected Facts:

- Incompatible or outdated versions of Common Vulnerabilities and Exposures (CVE) references.

2. Automated Security Testing Systems
The application uses industry standard security testing tools together with libraries to effectively obtain facts and data.

- Nmap serves as a networking tool that performs host discovery together with port scanning and network mapping functions.
- The system uses OpenVAS to execute comprehensive vulnerability scans which detect security problems including incorrect configurations and outdated software as well as known vulnerabilities.
- Python scripts based on Python language perform automated security assessments and collect data through security checks.

3. Data Collection and Analysis

Log File Analysis - Analyzes system logs to identify suspicious events, unauthorized access attempts, and potential breaches.
Packet Inspection - Examines network traffic at the packet level to identify anomalies and signs of an attack.
Risk Prioritization - Assigns severity levels to detected vulnerabilities based on predefined security criteria.

How it works:

- System or application logs are collected and analyzed to identify anomalies (e.g., repeated failed login attempts, unexpected service restarts).
- Can be integrated with Security Information and Event Management (SIEM) solutions if necessary.

Collected facts:

- Login attempts (both successes and failures)
- Changes in system services or user privileges
- Timeline of events that may highlight potential breaches

4. Automated reporting of collected facts
The software transforms unstructured scan results together with configurations into an easy-to-use PDF report

How it works:
The system gathers information about background scan results together with vulnerability data and configuration check results.
The FPDF library of Python automates report structure creation to produce readable documents which contain the following elements:

- List of discovered hosts and services
- Severity ratings of vulnerabilities
- Suggested remediation steps

5. Network and host scanning

The goal is to discover all hosts contained within specified network areas while determining their basic characteristics.

Through the utilization of tools such as Nmap the network segment receives a "ping" scanning operation to reveal existing hosts. The device discovery method reveals servers among other hardware components which require scanning.
Information collected:

- IP addresses of active hosts
- Port scanning and service enumeration.

The purpose revolves around discovering active TCP and UDP ports through which services execute on host systems.

Nmap alongside other similar port scanners exchange connection requests towards typical ports to reveal which ports provide responses. The port responds to the scanner which then attempts to determine which service from web server to database to SSH is active alongside its running version.
Information collected:

- Open/closed/filtered ports
- Service types.
- The discovery tools return information about running services through their service versions expressed as Apache HTTP Server 2.4.54.

2.2 Existing System

Current security assessment systems operate through different security tools targeting unique objectives. Security assessment systems deliver thorough checks but create multiple problems in the security framework.

1. Nmap
The open source tool Nmap exists primarily to perform both network discovery and port scanning functions. Nmap aids in detecting open TCP and UDP ports that allow port scanning on an intended target device. Service enumeration identifies active services through port inspections to discover the running programs HTTP and SSH and detects their version numbers.

The Python libraries python-nmap use Nmap application as back-end automation to generate network discovery reports about target hosts and services and provide unified reporting through automated input of raw scan results such as open ports and service banners.

2. OpenVAS
OpenVAS delivers complete vulnerability checks for known software flaws through its open source framework structure. OpenVAS represents an open source solution when customers seek an alternative choice to commercial tools like Nessus.
The vulnerability database remains constantly updated with all available latest CVEs.

The OpenVAS API supports basic security testing via its application programming interface with automated checks alongside CLI command execution to find potential vulnerabilities. OpenVAS results form the core data evaluation for final audit reports.

4. Nessus
The proprietary vulnerability scanner Nessus provides extensive vulnerability coverage through its large database of operating system and network device plugins while allowing users to refine and integrate its built-in reporting capabilities.

Organizations that possess Nessus licenses can choose to integrate the security testing application which will merge results into a project's automated final report while enabling a comprehensive validation of vulnerabilities through open source scanner comparisons (e.g. OpenVAS).

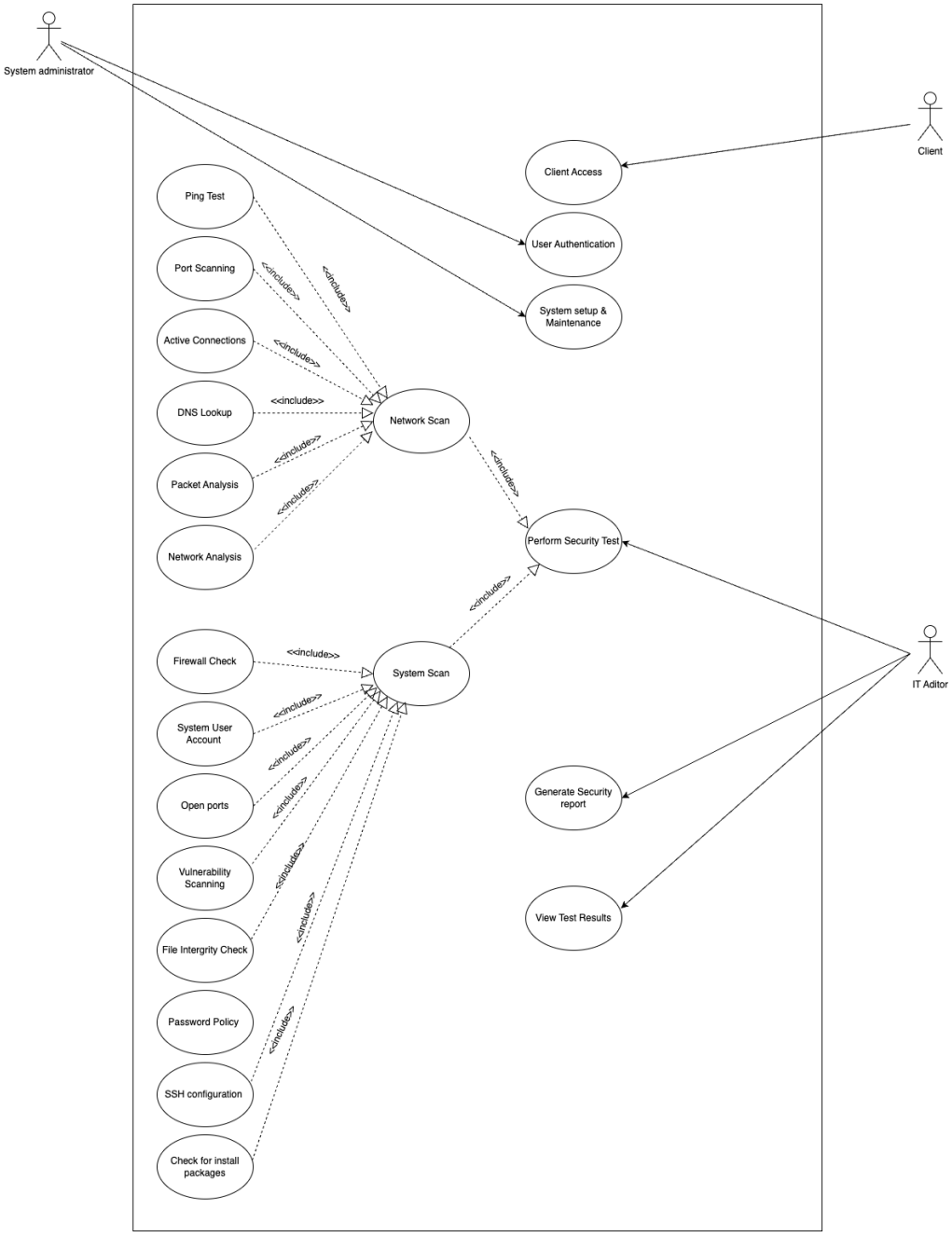
5. Other,
Wireshark
Wireshark functions as a network traffic capture tool even though it is not a dedicated vulnerability scanner but it reveals unexpected or suspicious network communications.

FPDF or similar PDF generation libraries
Through Python-based libraries the project extracts report data from initial tool results and converts it into structured professional PDF format.

These systems are important because,
The port enumeration and network scanning capabilities of Nmap work well together while the tool lacks any extensive vulnerability detection capabilities. OpenVAS and Nessus excel in providing detailed vulnerability

databases that can sort issues based on their severity values. A basic security testing application obtains comprehensive network and system and web vulnerability coverage when its tools are used together.
Comprehensive reporting: Everything from host discovery to high-level vulnerability summaries in a final report.
Organization and combination of data obtained from diverse systems enable identification of exposed ports along with their corresponding business ramifications and severity levels.

2.3 Use case diagram



2.4 Drawbacks of the existing system

Within the project proposal readers will find an extensive breakdown regarding the boundaries of current vulnerability-scanning alongside auditing systems.

Time-consuming manual processes:

Auditors currently need to execute different tools consecutively through their existing systems since port scanning tools require manual succession with other vulnerability scanners. Manually integrated results from security tests need to be combined into one end product. The manual process of data consolidation during audits brings an increased chance of human mistakes while making the audit process exceedingly slower.

Reporting solutions that are not integrated with each other and not automated:
Professional vulnerability scanners deliver detailed information but their exports generate unprocessed or partially processed data. Auditors dedicate substantial time to convert raw outputs from professional vulnerability scanners into reports that satisfy clients' needs. The ability to use scripts for automation exists yet many organizations lack the capabilities and technical know-how to develop their own programmed connections. Auditors waste numerous hours on scripting to process and normalize scan data because basic automated report generation capabilities are absent from their scanner tools.

Potential Gaps in Coverage
Multiple distinct specialized tools selected by auditors for network scans and system configurations and web application testing can result in protection gaps that individual scanners may fail to detect. A single scanning tool may perform well within open-port detection while lacking capability in misconfiguration assessment which creates incomplete system security visualization results.

Human Error and Inconsistency
The need for manual intervention throughout multiple steps in existing scanning/reporting methods creates opportunities for human mistakes which surface most prominently during raw result consolidation and vulnerability priority determination. The use of inconsistent report formatting between different clients or projects hinders overall tracking of progress as well as hinders result comparison across projects.

Chapter 03 - Requirements Specification

3.1 Functional Requirements

The focus is on the basic features and behaviors that the system must provide to achieve project goals. These requirements are grouped into the following key areas.

3.1.1 Configuring and starting a scan

The system allows an auditor to specify scan parameters. For example, the target IP range, port range, protocols, or vulnerable plug-ins can be specified. The goal is to give auditors fine-grained control over the devices, networks, or applications to be assessed.

A. Select the IP range or specific system components to select the scope of the scan.
Enter the IP range or specific system components to be scanned.
Enter the ports to be scanned.

B. Select the scan types.
The application is useful for multiple security tests that can be run individually or in combination.
Identifies vulnerable open ports.
Checks for known vulnerabilities in applications, software versions, and system configurations
Reviews security settings, firewall rules, and system hardening measures.
Prevents unnecessary scans on large or partitioned networks, speeding up the process.

Start Scan **:-**
After configuring the configurations, auditors can start scanning.

A. Launch Scan
Initiates the scan through the user interface.

B. Monitor Scan
Provides updates for scans that are running in the application.
Auditors can pause, resume, or cancel ongoing scans.

C. Handle Scan Results
Once complete, the scan generates a detailed security report.
Allows users to export findings as a PDF report.

Run Automated Scan -
The system integrates with existing SIEM tools for further analysis.

Benefits include efficient manual effort during security audits and multiple simultaneous tests.
Scheduled scanning and reporting reduces human error in identifying vulnerabilities.

3.1.2 Vulnerability scanning

Multiple security tests are run, reducing audit time and providing a comprehensive assessment of the target environment

Network and application-level scanning -
The system should scan network hosts and related applications for security vulnerabilities. This includes identifying open ports, identifying services and versions, and checking for outdated software or misconfigurations.

An auditor selects a target IP range to scan and checks each host for active services, their versions, and any known vulnerabilities associated with those versions.

Configurable scan types -
The system should allow auditors to choose different scanning methods based on their needs.
An auditor selects a quick port scan to verify which systems are alive, and then performs a full vulnerability scan that includes searches for known CVEs.

Real-time scan progress and status

- The system will display real-time progress for scans, such as percentage completed, time remaining, and critical findings.
- During a large-scale network audit, the system updates the auditor every few seconds with the number of hosts completed and the critical vulnerabilities found so far.
- Progress updates appear without having to refresh the page.
- The advantage of this is that it helps auditors manage their time and prioritize urgent vulnerabilities before a full scan is complete.

3.1.3 Automatic report generation

Comprehensive report creation -
A scan should be completed and the system should provide a complete audit report. The report includes risk summaries, severity levels, and recommended remediation steps.

Once a company's entire subnet is scanned, a PDF is automatically generated summarizing high-risk vulnerabilities in one area and low-risk issues in another. The advantage of this is that consistent, standardized reporting eliminates repetitive manual tasks for auditors.

Export and Sharing -
The system should export reports in standard formats and allow sharing via email or a secure link.
An auditor generates a PDF and immediately emails it to the client's IT manager for review.
The exported file format maintains consistency. This makes it easier for auditors to distribute findings and allows stakeholders to review them immediately.

3.1.4 Performance and Scalability

- The system should handle scans without crashing.
- It should manage resources efficiently when performing large scans across multiple subnets.
- Many real-world networks have many hosts and services, and the system needs to be able to respond under stress.

3.1.5 Reliability and Reporting Accuracy

- The system should verify results with external vulnerability databases.
- Where applicable, the application can cross-check known vulnerabilities for improved accuracy.
- The system generates consistent and accurate reports.
- Scan results should match the actual vulnerabilities in the environment. The modeling should be consistent across repeated use.

3.2 Non-Functional Requirements

3.2.1 Performance Requirements

Concurrent Scan Throughput -
The system provides parallel running capabilities for vulnerability scans that maintain both high speed and responsiveness. Running scans over networks with 50–100 hosts should complete their process within less than 2 hours when standard network conditions prevail.

Data Processing Speed -
The system needs to manage at least 1000 recorded vulnerabilities from individual scans without delays between them. The system needs to generate complete reports about at least 1,000 detected vulnerabilities in each scan run during processing time.

2. Reliability and Availability Requirements
Generation Time -
Between the scheduled audit period the application must operate with minimal service interruptions.

The system should resume operations within 5 to 6 minutes when it unexpectedly shuts down after recording the latest scan state.

Fault tolerance -
Errors that cause scans to fail should trigger application logging which enables users to either begin a scanned anew or resume their work from the point of failure. The solution needs to report clearly what results were obtained instead of presenting blank results whenever essential checks fail to execute.

Error recovery -
The system must attempt automatic recovery after it detects failure conditions.
Auditors must document all incomplete scans so they can refer to them in the future.

3.2.2 Security requirements

Secure data storage -
The encrypted storage of scan results together with user credentials and configuration details should be a part of the framework. For example, AES-256.
Secure protocols need implementation for all network connections to run between back-end modules and clients as well as between server systems and clients. For example, TLS/HTTPS
API endpoints that handle scan management must use authentication tokens together with secure session cookies to stop unauthorized access.

3.2.3 Usability requirements

Ease of use -

The user interface should guide non-technical or moderately technical auditors through the process. From selecting scan targets to generating reports, demonstrate with minimal complexity.

3.2.4 Maintainability

Modular architecture -

A modular application design provides the application core with loosely coupled components that enable module updates independently from other parts of the system. Plugins and third-party security solutions can be integrated as new modules to the system design.

Configuration Management -

A system of centralized version management exists for all configuration files to enable change tracking. Specification of scanning parameters and user preferences along with licensing details takes place in the system.

All configurations need to be differentiated between default settings and user-modified options by the system design.

3.3 Hardware / Software Requirements

3.3.1 Hardware requirements required for this project:

Test machine : A stable network connection is required to perform port scans, vulnerability scans, etc.

Test network environment:Switches, routers, or virtual network interfaces to simulate real-world environments.

Test servers : To perform a system check on a Linux operating system.

3.3.2 Software requirements required for this project:

1. Operating System
   - Host OS: Linux-based distribution.
   - Many security-oriented tools have native Linux support and can be easily configured on Linux.

2. Programming Languages

Python (3.x) is the main language for background logic for scanning, data processing, and report generation. The main language used for background logic is pip. Pip is available for dependency management.

Python libraries used:

- Two Nmap-specific Python libraries enable interaction with Nmap.
- Our system uses FPDF for creating automatic PDF reports.
- The programming package includes network functions through sockets and allows communication with the operating system using os and subprocess modules.

3. Security and Scanning Tools
   - The Nmap tool is essential for port scanning, service/version detection.
   - Optional tools include Wireshark (for packet analysis and troubleshooting), Metasploit (for deep penetration testing scenarios - to verify vulnerabilities you've found it primarily).

4. Security and Access Control
   - If multiple auditors are using the tool, you may need authentication and role-based access.
   - In a firewall configuration, allow outbound scan traffic on the host operating system.

3.4 Networking Requirements (Optional)

Systems require internet access. Security scans run in test conditions find all system weaknesses that would not impact operating systems during production.

3.4.1 Network Scanning Capabilities

The system is capable of performing network scans across a variety of environments, including local area networks (LANs), wide area networks (WANs) but the following network scanning features are essential:

IP Range Scanning-
This scans multiple IP addresses or an entire subnet to identify active hosts and open ports.

Port Scanning -
Scans open, closed, and filtered ports on a system to identify potential vulnerabilities.
Finds running services and assesses whether they are vulnerable, for example, outdated SSH or FTP servers.

Example: Open ports such as port - 22 (SSH), port - 80 (HTTP), port - 443 (HTTPS), and port - 389 (RDP).

Service and version detection -
Testing needs to identify the exact service versions that run the application to check for insecure older versions. Secure your computer by using both firewall and IDS evasion methods.
A scanning technique divides network packets into segments to escape particular firewall protections and study security status.
An IDS/IPS system finds attack signals to discover if it recognizes scanning efforts.

Recommended network components-
Network segmentation controls need testing software to connect with diverse network areas. Adjust the testing area into separate parts to check the system's functioning with different security barriers. Basic system functions to connect and direct scans across multiple subnets should be added. This approach makes it possible to test networks of different scale to find out scan effectiveness and speed.

3.4.2 Identifying Network Security Processes.

These security steps need to be put in place to stop new security flaws from appearing in the application. The system uses SSL/TLS to protect scan results sent through networks and connects to remote databases through VPN or SSH secure connections.

Network scans can create so many connection problems that control methods will reduce this issue.
Our system provides advanced settings to both manage scan speed and request numbers so networks stay protected from overloads.

Chapter 04 Feasibility Study

4.1 Operational Feasibility

The basic security testing application's fit with workflow and suitability for IT auditors and security professional requirements represents operational feasibility. The successful implementation of this solution requires evaluation of user adaptation needs alongside deployment situation analysis and training demands together with risk analysis assessment. Operational feasibility assessment has been conducted because this terminal-based application contains no user interface or database system.

4.1.1 Suitability for IT auditors and security professionals

The security testing application performs seamless scans through its basic design which makes it highly suitable for IT auditors. It is suitable for:

IT auditors and cybersecurity consultants:
Security audit automation remains the main purpose of this solution. Consultants obtain complete risk reports from this application to analyse security measures effectively.

Enterprise security teams (SOC analysts and IT administrators):
The application provides continuous security monitoring as one of its capabilities. The system performs scheduled scans which connect smoothly with current SIEM infrastructures to minimize human intervention.

4.1.2 How factors impact operational viability in detail

1. Administrator access requirements

IT auditors typically require administrator-level privileges/permissions to run extensive vulnerability scans and access critical system configurations. System-level security scans can be accessed using sudo/root.

For example, certain scans, such as port scans, vulnerability scans, or obtaining system logs, may require administrative privileges.

Auditors should have defined permissions that balance operational needs and security constraints.

2. Ease of use for IT auditors

Since the project is based on the Command Line Interface, simplicity in using terminal commands is essential. Commands should be easy to remember, and parameters should be clear and concise.
Providing a complete help page or manual documentation is important for ease of adoption. This is what this command (--help) is used for.
The application should be easy for auditors to quickly understand the command syntax, parameters, and expected outputs.

3. Level of Automation

Your tool should automate important factors such as vulnerability scanning, port analysis, and system configuration checks.
Automation reduces manual intervention by auditors, increases accuracy, and reduces audit time.
After each scan, reports should be automatically generated in this project, clearly describing the vulnerabilities, vulnerability levels, and remediation proposals.

4. Compatibility with auditor workflow

The system should align well with the auditors' existing workflow to ensure seamless integration into audit processes.
The generated reports are exported to a common format such as PDF, CSV, or JSON for easy inclusion in server deployments.
Integration with widely used audit standards and frameworks (ISO 27001, NIST, CIS benchmarks) ensures relevance and usability within current audit methodologies.

5. Data security and confidentiality

As automated scans produce reports containing sensitive risk information, data security is therefore important. These outputs should be managed securely to prevent unauthorized disclosure and procedures should be clearly outlined.

6. System Scalability

The system needs to handle security checks on both small host systems and extensive corporate network scans smoothly.

The software needs to maintain its operating capacity while performing numerous security audits simultaneously.

Scalability makes the application flexible by handling different audit assignments that emerge across various environments.

7. How WPA/WPA2 Testing Fits into the Project:

IT auditors require efficient tools to evaluate wireless network security therefore WPA/WPA2 authentication testing functions precisely within their job responsibilities.

Saves time and effort:

Security assessments of WPA/WPA2 through automation decrease human labor requirements which brings faster and more trustworthy audits.

The application design should maintain user-friendliness which enables non-specialist IT auditors to execute detailed security testing procedures.

This system passes operational feasibility because IT auditors can conduct WPA/WPA2 network assessments through the application without encountering difficulty.

4.2 Technical Feasibility

This feasibility report discusses the technical feasibility of the basic security testing application, the auditor's technical competence, system compatibility, compliance with security standards, system resource utilization, and how to technically integrate existing security tools.

It further explores whether the project can be developed using existing technologies, tools, and expertise.

1. Auditor's Technical Competency

IT auditors should have basic proficiency in the Linux operating system.

For example, they should have knowledge of basic Linux commands such as sudo, grep, cat, nano, chmod, etc.

Basic understanding of networking principles such as TCP/IP protocols, open/close ports, firewall rules, etc.

The application should have extensive reading, including clear command examples, manuals, and sample scripts, to bridge the proficiency gaps in cybersecurity tools such as Nmap, OpenVAS, and basic penetration testing techniques.

2. System Compatibility (Server Systems)

Since the application is explicitly Linux-based, the clients should have a native Linux environment.

Alternative methods such as live USB distributions or Docker images for non-Linux environments allow auditors to easily run the application.

3. System Performance and Scalability

The application needs to execute many parallel scans without overburdening system resources including CPU and memory. System reliability under various load conditions becomes more certain through regular testing of large network systems.

The application needs optimized Python scripts to maintain system resource efficiency for big-scale scanning operations.

4. Compliance with security standards

The reporting framework should implement standards including ISO 27001 and GDPR and NIST to provide more reliable and meaningful results. The security assessment of auditors depends on standards-compliant reports for their clients to demonstrate both relevance and security compliance needs.

Standardized reports enable client organizations to understand issues easily and implement necessary remedial steps.

5. System resource utilization
The terminal-based application should optimize its resource utilization to prevent straining system resources. Check if the application maintains an efficient operational state regardless of system hardware capacities.

6. Impact of WPA/WPA2 testing.
Your application can integrate both Scapy and Wireshark tools as Wireless Packet Analysis Tools to perform wireless network authentication analysis. The implementation of libraries that establish network interface connection for performing WPA/WPA2 authentication procedures.

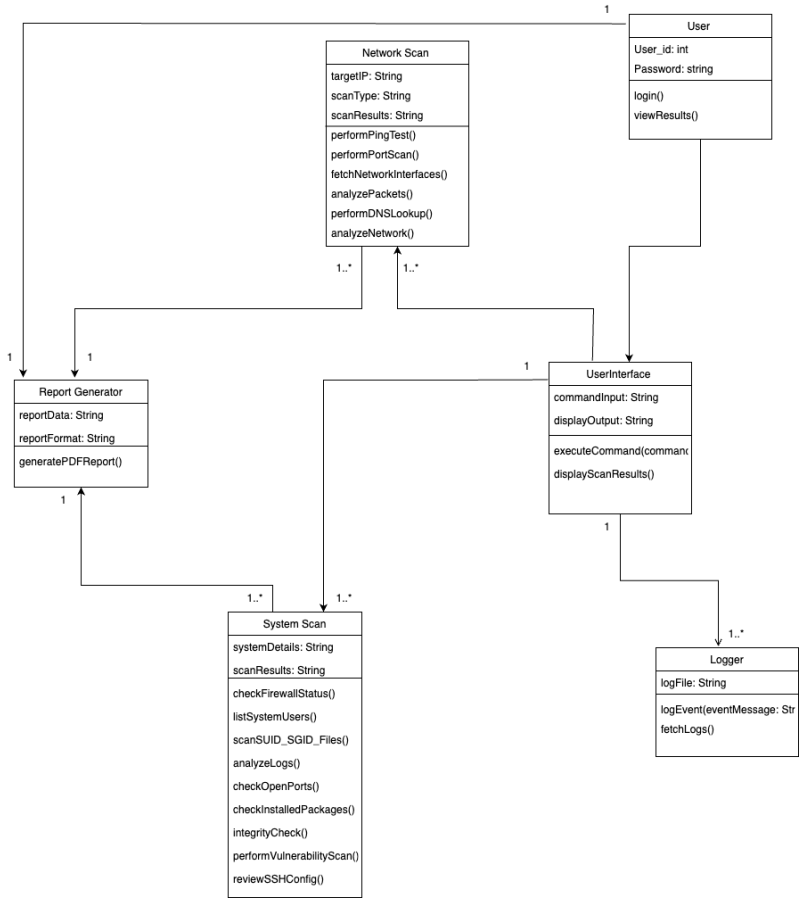
Since the project depends on peripherals the system needs to process WPA/WPA2 scans using available hardware without additional specifications. Through available resources of programming expertise and necessary systems and appropriate technologies the implementation of WPA/WPA2 security testing proves possible for the project.

4.3 Outline Budget

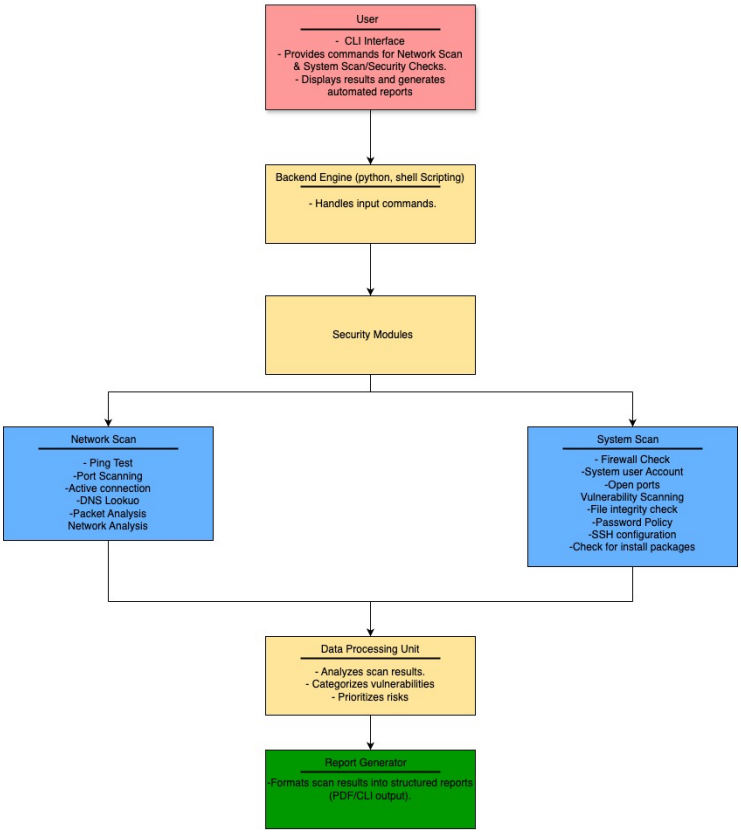
No Budget Allocation for this Project. No Budget Allocation for the Project due to use of open-source tools and technologies, no hardware costs, standalone study/research-based project, and no staff or external development costs

Chapter 04 System Architecture

5.1 Class Diagram of Proposed System



5.2 High-level Architectural Diagram



Chapter 06 Development Tools and Technologies

6.1 Development Methodology

The basic security testing application is developed following the Agile software development methodology. Agile software is chosen because of its iterative, flexible, and feedback-driven nature, which means continuous testing and improvement throughout the entire process. If new cybersecurity threats emerge, the application can quickly adapt and incorporate the necessary updates.

6.1.1 why Agile software is chosen

because,

- Continuous testing can identify bugs and improve security.
- Continuous iterations reduce development time due to rapid development cycles
- Improved security allows for frequent updates to allow for continuous security improvements.

6.1.2 How Agile is involved in this project

1. Development process and testing
The project is divided into manageable modules:

- Vulnerability scanning
- WPA/WPA2 security testing
- Automatic report generation etc.

Each module is developed and tested here. This minimizes the likelihood of major system failures.
Here, the system is verified to be functional and useful at every stage of development.

2. Regular testing and evaluation
Cybersecurity tools should be tested to ensure accuracy and reliability.

3. Alignment with IT audit processes
IT audits require flexibility in security over the network, infrastructure.
Agile allows the tool to be customized and refined based on feedback from potential users.
If auditors need additional scanning features, they can change the system without planning.

6.1.3 How Agile connects to other aspects of the project

Project aspect Agile development impact
Features using WPA3 in future updates allow for new security testing to be added gradually.
Daily testing helps to quickly identify bugs, improve system stability. Agile allows for easy changes.

The development of the basic security testing application uses the agile development methodology. Since security testing requires flexibility, frequent testing, and incremental improvements, Agile is the best fit for the basic security testing application.

6.2 Programming Languages and Tools

The basic security testing application is developed using a combination of secure programming languages and tools. These tools have been selected to ensure that the application is efficient, scalable, and secure while supporting cross-platform deployment.

6.2.1 Programming languages used

1. Shell Scripting:
Shell scripting is a useful tool for automating routine tasks and system-level operations in Linux-based environments. It is suitable for system and network security audits.
It supports ping tests, firewall status checks, system user account verification, and basic file system analysis, such as finding SUID/SGID files.

Advantages:
Efficiency: Perform repetitive security checks such as active connection detection and firewall rule validation.
Integration and flexibility: Easily integrates with Linux/MacOS command line utilities.
For example, netstat, iptables, ping, ifconfig.

No extensive additional software installations required, lightweight terminal-based application.

- Firewall rule checking (iptables -L)
- Active network connections detection (netstat -tunap)
- System user enumeration (cat /etc/passwd)
- SUID/SGID file checks (find command) etc.

2. Python:
Python is widely used in cybersecurity due to its ease of use, extensive library support, and readability. This makes it ideal for complex, automated security applications and vulnerability scanning.

System Interaction:
Executes shell commands within Python scripts for checks such as SSH configuration, firewall status, SUID/SGID file checks, and system account validations.

6.2.2 How programming languages and tools are involved in the project:

Python:
Quickly and accurately automate complex vulnerability scanning tasks through tools like Nmap and Python scripts.

Shell scripting:
Efficiently performs scans such as ping tests, port checks, and firewall status assessments.

Easy-to-use interface:
Python scripts handle the backend complexity, presenting simple output in an easy-to-read format directly on the terminal. Simple shell commands are used for terminal-based, intuitive interaction.

Simultaneous execution of multiple security tests:
Python's concurrent modules enable the execution of multiple scans in parallel, significantly reducing time.

Automated PDF report generation:
Provides professional-looking, automated PDF report generation directly from Python scripts, saving auditors significant documentation time.

Shell scripting:
It is possible to preprocess logs and generate basic text reports before generating detailed PDFs via Python.

6.2.3 Tools used

Security Testing Tools:
1. Nmap:
Important for the project as it is widely used for network exploration, security scanning, port scanning and vulnerability detection.
Helps auditors quickly identify open ports, services running on those ports, service versions and potential vulnerabilities.

2. Wireshark:
tool for packet analysis and deep packet inspection. Useful for validating network traffic during analysis stages or for detailed packet-level inspection.

System and Network Analysis Tools (Built-in Terminal Commands):

- netstat: Used to check for active network connections and open ports.
- Nmap: Scan the entire network for active devices.
- Tcpdump: Capture and analyze network packets.
- nmap & nectar: Discover open ports on a target host.
- iptables (Linux) / pfctl (MacOS): To check firewall rules and firewall configuration status.
- ping: To check network connectivity and latency.
- nslookup/dig: Performs DNS lookups, supports DNS resolution and domain analysis.
- ifconfig / ip: Provides detailed network interface details such as IP addresses, subnet masks, and interface status.
- sudo, find: To identify vulnerable SUID/SGID files on Unix/Linux systems.

Automated reporting tools:
FPDF: Automates the creation of PDF reports directly in the Python environment, easily producing consistent and professional audit documentation.

Development tools:
Visual Studio Code: The recommended integrated development environment (IDE) for Python and shell scripting, offering plugins and debugging tools that streamline coding and development workflows.

6.3 Third Party Components and Libraries

Third-party build options help software developers expand product features while making processes easier and more dependable. These core third-party components and libraries need assessment during your security testing process:

1. Nmap
Used for network exploration, port scanning, and vulnerability detection to scan open ports, discover active hosts on the network, identify services running on those ports, and identify vulnerabilities related to network services and configurations.
Integrated using the python-nmap library.

python-nmap:
A Python-based integrated library that allows you to seamlessly execute Nmap commands from within Python scripts.
Simplifies the use and automation of Nmap directly from Python scripts.

3. Active Directory
Can be used to analyze and audit security configurations in corporate network environments that use Active Directory (AD).
Enables assessment of user privileges, group policies, and system security postures in Windows-based IT environments. Useful when auditing environments with complex Windows infrastructure.

4. FPDF (Python Library - PyFPDF)
A Python library used to programmatically generate PDF documents.
Provides simple methods for generating professionally formatted PDF reports directly in Python.
Improves the efficiency of audit documentation, eliminating manual reporting errors and inconsistencies.

5. OpenSSL/OpenSSH
Verifies and tests SSL/TLS and SSH configurations to assess encryption and security practices.
Essential industry-standard tools for validating and analysing secure communication protocols and authentication mechanisms.
Provides reliable methods for ensuring secure configurations and identifying cryptographic vulnerabilities.

6. Telnet
A quick way to quickly assess the status of TCP ports by attempting direct connections and verifying the status of the port and connection (open/closed).
Easily scriptable, integrates directly with Bash and Python scripts.

6.4 Algorithms

The Basic Security Testing platform uses its main algorithm to check the security health of computers through automatic testing and scoring. The scoring system forms the essential part of the security analysis process by delivering accurate results.

How the algorithm works:

1. Startup and data collection
The algorithm starts by running several predefined security tests simultaneously.
among which,
are network scans and system security checks.

2. Risk severity assessment
Each vulnerability discovered is classified according to its severity level:
This scoring is aligned with industry-standard vulnerability scoring systems such as the Common Vulnerability Scoring System (CVSS).

3. Security score calculation
The algorithm calculates an average security score using a weighted average methodology. The mathematical representation is as follows:
The numerator is the severity score added from all identified vulnerabilities.
The denominator is the highest severity score when all tested components are assumed to have critical vulnerabilities.

4. Result Classification
Based on the calculated score, the algorithm classifies the security status.

5. Automatic Report Generation
Post-scoring, the algorithm triggers an automatic report generation process.

- Detailed lists of identified vulnerabilities.
- Severity scores assigned to each vulnerability.
- Overall security assessment and recommendations for mitigation.

This automated PDF reporting feature significantly reduces manual efforts, ensures compliance, and speeds up decision-making processes.

Benefits:

- Time Efficiency: Concurrent security testing reduces audit time.
- Improved accuracy: Eliminates human errors inherent in manual testing.
- User Accessibility: Provides reports that are easy to understand for auditors with varying technical expertise.

Technology Used:

- Python for backend logic
- Integration with security tools
- Automated reporting using Python's FPDF library

Chapter 07 Discussion

Interim Report Overview

The current report shows our work to create the basic security testing system. This initiative creates a total endpoint security test solution that IT auditors and security experts can use. The project uses automation to handle necessary system scans and reporting which enhances network and system security tests. This document explains the project's system analysis techniques while presenting required specifications, feasibility assessments, system design, working tools, and development tools.

Report Summary

This document starts by describing the project's purpose in cybersecurity auditing. It finds key challenges with traditional methods of auditing and lists the difficulties caused by manual testing tools Nmap and OpenVAS. The requirements analysis explains what the tool should do and how it should work plus standard requirements for usability, accuracy and speed. The feasibility study helps determine if an updated cybersecurity tool for auditing purposes can function properly with Python scripting and automated reporting tools during operations. The system design team picked tools that enhance scanning performance plus keep the system lightweight when running Linux OS.

Challenges Faced

- Library Installation Issues: Some devices had problems installing necessary libraries when such installations failed to match existing system specifications or when essential support files were missing.
- Limited Internet Access: When systems did not have Internet connectivity, they could not update their security features nor retrieve digital threat data for time-sensitive virus checks.
- Permission restrictions: The application needs administrative rights to do complete security assessments that involve port scanning and logging view access. The security settings in those systems were too strict and prevented the application from completing its tasks.
- Performance optimization: The program needed to perform several tasks at once without slowing down the system but using resources effectively presented a challenging obstacle.

Future Plane

- User-friendly interface: A better terminal-based software needs a user-friendly interface with clear commands that teaches users how to navigate it.

- Efficient database integration: Our system manages scan results and found flaws through a lightweight database for easier monitoring and evaluation.

- Improved reporting features: The tool now makes better security reports by showing precise vulnerability information along with risk grades and recommended actions.

- Cross-platform compatibility: Although primarily developed for Linux, future iterations are aiming to include compatibility with macOS and Windows for wider use.

- Inclusion of advanced security testing: Expanding testing features to include wireless security assessments (WPA/WPA2 scanning) and malware detection enhances the tool's auditing capabilities.

Reference

[1] S. Kamara, S. Fahmy, E. Schultz, F. Kerschbaum, and M. Frantzen, "Analysis of Vulnerabilities in Internet Firewalls," Mar. 2003. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0167404803003109

[2] E. C. Lo and M. Marchand, "SECURITY AUDIT: A CASE STUDY," Niagara Falls, ON, Canada, May 2004. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/1344989

[3] K. D. Jadhav, "THE ROLE OF CYBER SECURITY AUDITS THE ROLE OF CYBER SECURITY AUDITS IN MANAGING COMPANY SYSTEMS AND APPLICATIONS," Jan. 2023. [Online]. Available: https://www.researchgate.net/publication/367559332_THE_ROLE_OF_CYBER_SECURITY_AUDITS

[4] G. Murali, M. Pranavi, Y. Navateja, and K. Bhargavi, "NETWORK SECURITY SCANNER," 2011. [Online]. Available: https://d1wqtxts1xzle7.cloudfront.net/90636070/2af5cc555cae555660bb4226fab380a43594-libre.pdf?1662272260=&response-content-disposition=inline%3B+filename%3DNetwork_security_scanner.pdf&Expires=1730206193&Signature=JXcwMhMP5aGKSFIRlA03XQmd--AeCJvEeeji9C3Sm76svApm~d-yaAif7MZnAU9SirtkTktlpyJ8ZjUrgmdlzFDyh50YpRfAC2ILLx7suGETBBY0eJhX5Gg2iZ6uwuxYutWHZfUHSJgL1HmpGJfTZAM7rkieuNaZYohC8MZoNLCkdyyt3b1YySaB7TBSzaSUwYLsp33k1u4y~NHyUas~FfU-ry8k4jjEGpFH~pcIusDMOdsBh6OWyFENMjPeXENllyyX0OvWA8P4eC5stKRlHBO1nWQxRA3di94Y1J9y24ck8EAJdgLyVD650rmeOwgbBj3IKacbakgDgFhp1SPluQ__&Key-Pair-Id=APKAJLOHF5GGSLRBV4ZA

[5] A.-M. Suduc, M. Bîzoi, and F. Gheorghe FILIP, "Audit for Information Systems Security," 2010. [Online]. Available: https://citeseerx.ist.psu.edu/document?repid=rep1&type=pdf&doi=8fe27c55f5298a4a67a8204f4596b3609440556e

[6] A. Aarthy Devi, A. K. Mohan, and M. Sethumadhavan, "Wireless Security Auditing: Attack Vectors and Mitigation Strategies," in *Procedia Computer Science*, Elsevier B.V., 2017, pp. 674–682. doi: 10.1016/j.procs.2017.09.153.

[7] Bayu Rima Aditya; Ridi Ferdiana; Paulus Insap Santosa, *Toward Modern IT Audit– Current Issues And Literature Review*. Yogyakarta, Indonesia: IEEE, 2018. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/8528627

[8] P. Boyanov, "A COMPREHENSIVE SCANNING FOR OPEN, CLOSED AND FILTERED PORTS IN THE COMPUTER SYSTEMS AND NETWORKS," *Original Contribution Journal scientific and applied research*, vol. 23, 2022.

[9] Jayant Gadge and Anish Anand Patil, *Port Scan Detection*. New Delhi, India: I E E E, 2008. Accessed: Feb. 02, 2009. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/4772622

[10] V. N. Sayankar and V. N. Sayankar, "A Review on Information Systems Audit," *Research J. Engineering and Tech*, vol. 4, no. 3, Sep. 2013, [Online]. Available: https://www.researchgate.net/publication/381653679_A_Review_on_Information_Systems_Audit

[11] S. Rao Vemula, “Automating Security Testing: Strategies for Vulnerability Scanning, Penetration Testing, and Compliance Checks,” *International Research Journal of Engineering and Technology (IRJET) e-International Research Journal of Engineering and Technology*, Jul. 2024, [Online]. Available: www.irjet.net

[12] S. Rao Vemula, “Automating Security Testing: Strategies for Vulnerability Scanning, Penetration Testing, and Compliance Checks,” *International Research Journal of Engineering and Technology (IRJET) e-International Research Journal of Engineering and Technology*, Jul. 2024, [Online]. Available: www.irjet.net

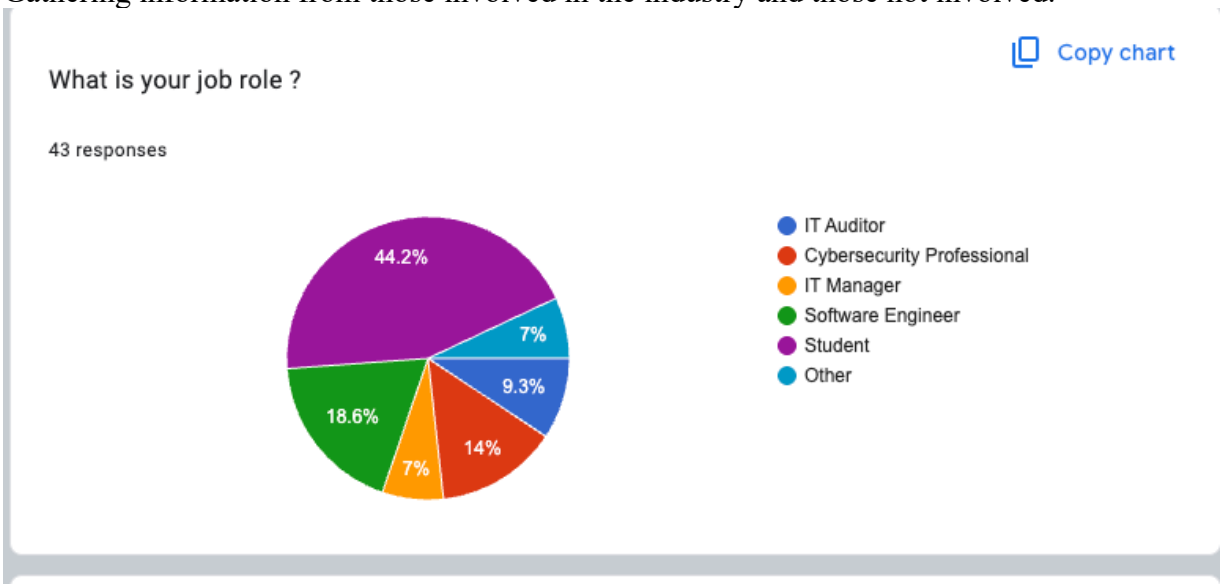
[13] J. P. Seara and C. Serrão, “Intelligent System for Automation of Security Audits (SIAAS),” *EAI Endorsed Transactions on Scalable Information Systems*, vol. 11, no. 1, Oct. 2024, doi: 10.4108/eetsis.3564.

[14] T. Vieira and C. Serrão, “Web Applications Security and Vulnerability Analysis Financial Web Applications Security Audit-A Case Study,” Dec. 2016. [Online]. Available: https://repositorio.iscte-iul.pt/bitstream/10071/12991/5/Web-Applications-Security-and-Vulnerability-Analysis-Financial-Web-Applications-Security-Audit%E2%80%93A-Case-Study.pdf

[15] A. Ziro, S. Toibayeva, S. Gnatyuk, A. Imanbayev, M. Iavich, and Z. Zhaybergenova, “Research of the Information Security Audit System in Organizations,” in *SIST 2023 - 2023 IEEE International Conference on Smart Information Systems and Technologies, Proceedings*, Astana, Kazakhstan: IEEE, May 2023, pp. 440–444. doi: 10.1109/SIST58284.2023.10223557.

Appendixes

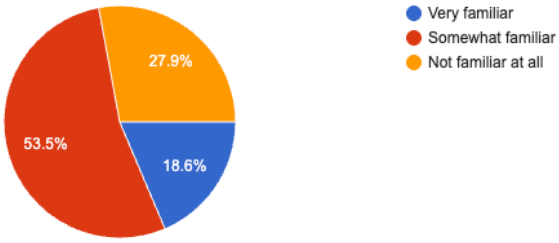
Gathering information from those involved in the industry and those not involved.



How familiar are you with IT auditing and security assessments?

Copy chart

43 responses

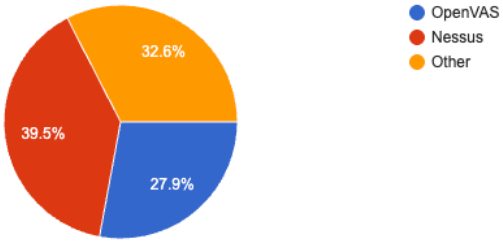


Questions for IT Auditors

What security tools do you currently use for vulnerability assessments?

Copy chart

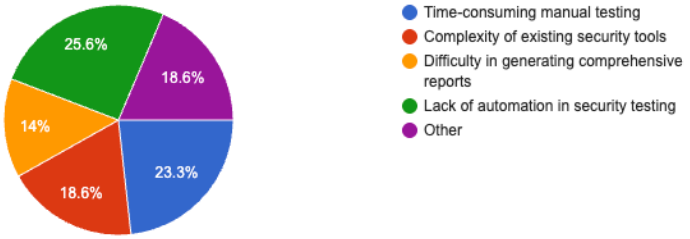
43 responses



What challenges do you face in IT auditing?

Copy chart

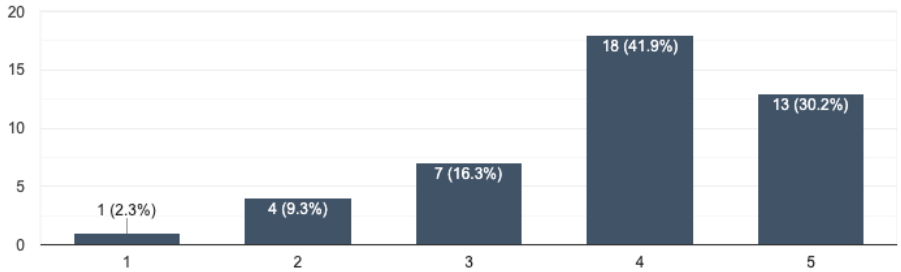
43 responses



How important is the security check to you?
(Scale: 1 - Not Important, 5 - Extremely Important)

Copy chart

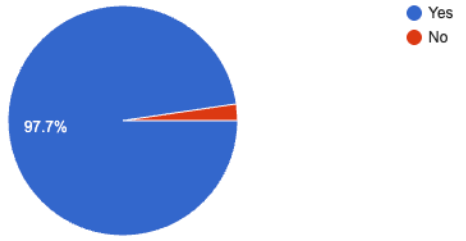
43 responses



Would you like to be able to perform security scans and generate reports through this application to benefit your work?

Copy chart

43 responses

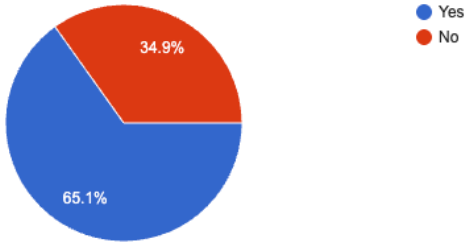


Questions for Non-Auditors

Have you ever conducted a security audit or vulnerability assessment?

Copy chart

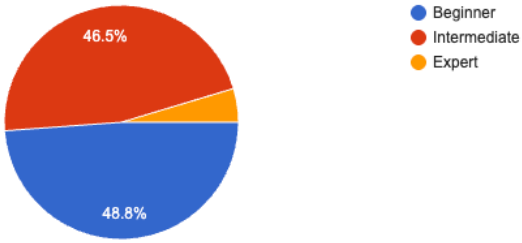
43 responses



What level of technical knowledge do you have in cybersecurity?

Copy chart

43 responses

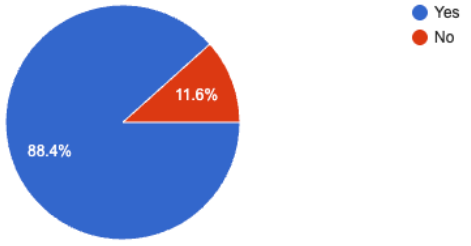


final gather

Would you be interested in testing or providing feedback on a security auditing tool?

Copy chart

43 responses



Do you have any additional suggestions or comments regarding the IT audit tool?

17 responses

Enable scans without requiring continuous internet access.

Make it compatible with windows, Linux

Scheduled Scanning Feature

I have not

Be aware of Security and Privacy Considerations

Multi-Factor Authentication, Audit Logging, Data Retention & Compliance

Enable scans without requiring continuous Internet access

create a user-friendly Interface & generate clear report out-put of this project

Ensure an intuitive design that requires minimal technical expertise.