

Comparative analysis of network vulnerability detection tools

Mateusz Zdunek*

Department of Computer Science, Lublin University of Technology, Nadbystrzycka 36B, 20-618 Lublin, Poland

Abstract

In times of growing global tensions, attacks on ICT infrastructure are becoming increasingly common. The field of IT systems security is becoming even more important. Various types of network vulnerability testing tools allow for the examination and improvement of the security of such systems. This article aims to compare and analyze selected network vulnerability testing tools. The systems that were examined are Nessus by Tenable, which is an enterprise-class tool, OpenVAS by Greenbone, which is an open-source project, and Nmap, a standard network mapping utility explicitly utilized in this study as a baseline tool.

Keywords: security; network security; network vulnerabilities; vulnerability detection tools; cyber security

*Corresponding author

Email address: s96751@pollub.edu.pl

Published under Creative Commons License (CC BY 4.0 Int.)

1. Introduction

Modern IT systems, including growing internal networks, are the basis of today's companies and public institutions. With the enlargement of IT structures, cybercrime grows, where most of it is based on financial benefits. Mandiant research shows that 55% of attacks in 2024 are motivated by profits [1]. Research shows that network-based attacks are becoming more frequent, for example, DDoS-type attacks increased by 46% in 2024 compared to 2023 [2]. Usage of vulnerability scanning tools should be implemented in every IT environment to ensure better security of said systems. Only in 2024, over 40 thousand new CVE common vulnerabilities and exposures were found [3]. This year was a record year in terms of the number of new vulnerabilities.

Despite the large number of tools available and the increased risk of cyber attacks in recent years, comparative studies evaluating the efficiency of commercial and open source scanners remain relatively limited. Therefore, the primary thesis posited in this paper is that effective detection of vulnerabilities in a computer network requires the use of various tools that have different functions and mechanisms of working. One tool cannot identify all classes of threats and security gaps. To verify this thesis, the following hypotheses were formulated. Automated vulnerability scanners discover a larger number of vulnerabilities than tools designed primarily for network scanning. Authenticated scanning significantly increases the detection rate compared to unauthenticated scanning. Execution time of the test depends on the utilized tool and the level of detail of the analysis. The main purpose of the article is to verify the thesis and hypotheses put forward through a comparative analysis of selected tools. The tools selected for this analysis are OpenVAS, Nessus, and Nmap.

2. Analysis of available solutions

The number of tools available on the market is large and growing every year. There is a wide range of commercial and free programs available. Usage of these

tools allows engineers to increase security in IT systems [4]. Choosing the right software that is tailored to the given needs allows engineers to work effectively to eliminate vulnerabilities in the tested environment. Tools can be divided into a few groups. The first group is a set of programs that map networks and allow for basic vulnerability hunting based on reporting software version or open ports [5]. The next one is a group of automated vulnerability scanner programs that are dedicated to checking for network security gaps. Those work automatically and are far more advanced in checking for vulnerabilities. Additionally, these use more advanced types of scanning, including authenticated scanning [4]. Programs can also be divided into two groups based on licensing models. There are free open-source tools and paid commercial ones (see Table 1).

Table 1: Summary of vulnerability scanning tools

Name	License type	Primary use	Location of use
Nessus	Commercial	Vulnerability management	Local / Cloud
OpenVAS	Free	Vulnerability scanning	Local
Nmap	Free	Network mapping	Local
Rapid7 InsightVM	Commercial	Risk management	Local / Cloud
Qualys VM	Commercial	Vulnerability management	Cloud
OpenSCAP	Free	Security baselines	Local
Acunetix	Commercial	Web apps scanner	Local / Cloud
GFI LanGuard	Commercial	Patch management	Local
SAINT Security Suite	Commercial	Vulnerability management	Local / Cloud
Retina	Commercial	Risk assessment	Local
Metasploit	Free / Commercial	Pentesting	Local
Wireshark	Free	Packet analysis	Local

Based on the analysis, a set of three tools was chosen for the comparison. The first one is Nessus, as the leading software in the commercial market that has full

technical support from its creators. Next is OpenVAS, the most popular representative of open-source tools, allowing for a comparison of a free scanner with a commercial one. The last one is Nmap, which is a network exploration tool rather than a dedicated vulnerability scanner. While Nmap Scripting Engine (NSE) enables it to do basic vulnerability hunting, Nmap was explicitly included in this research to serve as a baseline tool. This establishes a point of reference to evaluate whether standard network reconnaissance software can compete with specialized vulnerability management solutions. The selection of these three tools allows for the research of three different approaches to vulnerability detection. The diverse selection is crucial for the verification of the thesis and hypotheses put forward.

3. Previous Research

A review of existing literature reveals several comparative studies on network vulnerability scanners, with researchers primarily focusing on detection efficacy, usability, and performance metrics. Studies by Toyin and Chalvatzis demonstrate that commercial software, particularly Nessus, generally outperforms open source alternatives in terms of the total volume of detected vulnerabilities and precision [4, 6]. Furthermore, recent research by Alghamdi indicates that open-source tools often struggle with higher rates of false positives and false negatives, which can significantly increase overall security risks [7]. Regarding scanning speed, findings in the literature vary depending on the testing environment. For instance, Toyin observed that OpenVAS completed specific scans faster compared to Nessus, suggesting it as a viable option for rapid preliminary assessments [4]. On the other hand, Nmap, while traditionally classified as a network mapping tool, is frequently highlighted by Samera and Chaturvedi for its unparalleled speed, automation capabilities, and minimal resource footprint [5, 8]. However, it is rarely considered a direct competitor to comprehensive scanners in deep auditing scenarios. Despite these existing comparisons, a significant gap remains in the current literature. Many older studies do not account for modern operating systems such as Windows 11, nor do they evaluate the newer Greenbone Vulnerability Management engine introduced to OpenVAS. This clearly justifies the necessity for an updated, multi-criteria analysis using modern software architectures.

4. Research plan and assumptions

The research methodology ensures a rigorous evaluation of fundamentally different vulnerability management philosophies: Nessus as a commercial standard, OpenVAS as an open source tool, and Nmap with NSE network reconnaissance and as a baseline tool. The testbed utilized Windows Server 2019, Windows 11, and Metasploitable 2. Specific ground truth vulnerabilities were implemented to test network-level and local exploitation vectors, including non-standard services such as HTTP on port 8888, outdated local software such as Mozilla Firefox, VLC, 7-Zip, known backdoors

using vsftpd, and unpatched protocols, in this case, SMBv1.

To maintain mathematical integrity, the formal experiment followed a continuous four-phase procedure. The process began with environment provisioning, establishing an isolated NAT network, and Ground Truth setup. This was immediately followed by the execution of Discovery, Unauthenticated, and Authenticated scanning scenarios. Finally, a multi-criteria synthesis applied the SAW algorithm to calculate the final evaluation score.

5. Research environment

Research was conducted in a controlled laboratory isolated from the production environment to ensure security. This also allowed to test the tools on adjusted systems each time under unchanged conditions. The environment was built based on the main virtual machine from which the scan is performed, as well as other systems that are being targeted.

5.1. Virtual and network environment

Oracle VirtualBox was used as the virtualization software, which allowed the entire system to be created within a single physical machine. Using VirtualBox allowed to create a NAT network, which only allows for these systems to connect to the WAN and each other (see Figure 1). This setup was chosen to confirm that all of the security tools are up to date, and their vulnerability database is updated.

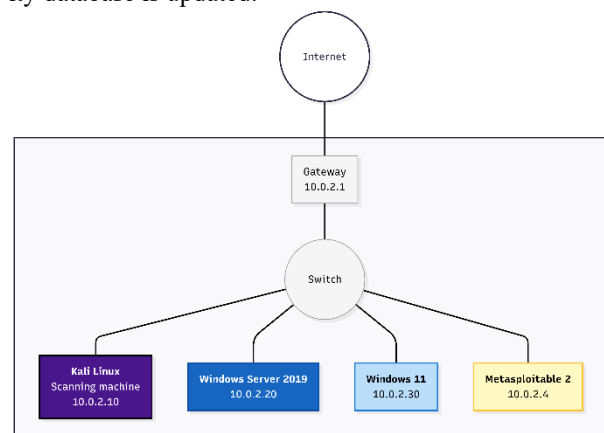


Figure 1: Test network topology

The test environment utilized a private IPv4 subnet configured as 10.0.2.0/24. The default gateway has been assigned to the address 10.0.2.1, while the DNS server address has been set to 8.8.8.8.

Three systems that were implemented to work in this virtual environment as a target for scanning tools were Windows Server 2019, Windows 11, and Metasploitable 2. The choice of these systems was dictated by the desire to test the tools in environments often used in production and non-commercial environments. In 2025, Windows had almost 28% of the market share of all operating systems, which was the second-best result in terms of all electronic devices. However, looking only at the computer segment, Windows has almost

70% market share, which positions it as the most popular operating system [6]. The Metasploitable 2 system is designed for cybersecurity specialists to conduct tests on it. It is designed to have a large number of vulnerabilities, old software versions, and web application vulnerabilities. The use of this type of system allowed for testing tools in an environment that is critically vulnerable to attacks. Windows Server 2019, on the other hand, is an enterprise system, so its use during testing increases the usefulness of the analysis.

The system used for scanning is Kali Linux. It is a Linux distribution commonly used in the field of computer security. This system will be used to install and run scanning programs.

5.2. Implemented security vulnerabilities

In the research methodology, specific security vulnerabilities were implemented on each of the tested systems. Each of them was selected to test different aspects of the scanners. The selection was made to test several criteria. The first one is to check the network scanning range. To test this parameter, the setup of an open HTTP service on port 8888 in Windows Server will be used. The usage of this type of gap will allow scanners to verify ports that are not commonly used [7]. Another criterion is testing vulnerability signature database of the scanners. This will be based on testing the old version of the software that has flaws. The software in question is VLC Media Player, 7-Zip, and Mozilla Firefox, which will be installed on Windows 11. Using this type of solution will allow to check the results of authenticated scanning in a more accurate way. Vulnerabilities in outdated software versions are very common, allowing for many attacks using them [8]. The third aspect of vulnerability implementation is testing detection mechanisms. Among other things, the Metasploitable 2 system has a built-in vulnerability in vsftpd version 2.3.4, which is responsible for the FTP protocol. Identifying this flaw will allow testing of whether the tools can accurately detect specific backdoors that should not be present in the services in question. Another criterion is a benchmark for all scanning applications by detecting the vulnerable SMBv1 protocol, which is commonly used in production environments. It will be implemented in Windows Server. This vulnerability is well known, and if the scanner does not detect it, it means that it has been configured incorrectly. Additionally, this will allow for comparison of the CVSS risk assessment system between reports generated after scanning. Additionally, accounts have been configured in Windows Server, Windows 11, and Metasploitable 2 systems with weak passwords. Another important aspect to evaluate is the detection of unencrypted protocols. These are FTP configured on Windows Server and Telnet set on Metasploitable 2. The Metasploitable 2 system itself has many different vulnerabilities, which will only be used to confirm the effectiveness of the tools. Vulnerabilities that research will focus on regarding Metasploitable 2 system are Java RMI and UnrealIRCd (see Table 2). A detailed summary of the vulnerabilities

specifically configured within the Windows environments is presented in Table 3.

Table 2: Metasploitable 2 targeted vulnerabilities

Application/port	Vulnerability name	CVE	Description
Vsftpd / 21	Backdoor Command Execution	2011-2523	Vulnerability in the FTP server code
UnrealIRCd / 6667	IRC Daemon Backdoor	2010-2075	Modified version of the IRC server containing a backdoor
Java RMI / 1099	Java RMI Insecure Default Config	2011-3556	Vulnerability allowing Remote Code Execution
Telnet / 23	Weak Credentials	n/a	Access using weak credentials

Table 3: Summary of vulnerabilities implemented in Windows systems

Application / Service	Version	CVE	Description
Mozilla Firefox	50.0	2016-9079	Remote code execution
VLC Media Player	2.0.4	2012-5855	Execution of malicious code when opening a multimedia file
7-Zip	16.04	2018-10115	Remote code execution when extracting an archive
System user	n/a	n/a	Verification of weak passwords
http, port 8888	n/a	n/a	Verification of non-standard ports
SMB	V1	2017-0144	A vulnerability allowing RCE
FTP, Telnet	n/a	n/a	Use of protocols without encryption

5.3. Selection and configuration of tools

The selection of tools for comparative analysis was based on a review of available programs and their different features. Scanners that were selected were deployed and configured on a fully updated Kali Linux system. To monitor and record resource utilization during the scanning process, the dstat monitoring tool was used. The metrics regarding CPU and RAM usage were gathered to calculate resource utilization.

5.3.1. Nmap configuration

Nmap version 7.98 was used in the Kali Linux environment. Before starting the research, the Nmap Scripting Engine (NSE) database was updated to ensure the availability of the most recent detection scripts and signatures.

5.3.2. OpenVAS configuration

The OpenVAS tool was installed and configured on the Kali Linux machine. To ensure the best results, all threat feeds, including network vulnerability tests (NVT), security content automation protocol (SCAP), and CERT data, were synchronized to the most current state.

5.3.3. Nessus configuration

The Nessus vulnerability scanner version 10.11.3, developed by Tenable, was deployed on the scanning machine utilizing a Nessus Essentials Plus license. Before testing this tool, the Nessus plugin database was fully synchronized with official repositories. Synchronization ensured that the scanner would work as best as possible.

6. Methodology

To verify the thesis and hypotheses, a comparative analysis of the selected tools was conducted. Discovery, Black box, and white box testing scenarios were developed, each focused on different aspects of vulnerability scanning. The final evaluation of the scanners was performed utilizing a multi-criteria decision-making model.

6.1. Comparative metrics

To accurately evaluate the vulnerability scanners, a set of metrics was defined. The multi-criteria assessment utilizing the simple additive weighting (SAW) method relies on four primary parameters. Additionally, two more metrics were introduced to analyze differences between specific scanning scenarios. Primary criteria for the SAW model are as follows:

- Targeted Detection Rate (WS) is the main metric that is used for evaluating the scanner's precision. Instead of measuring global detection capability, it calculates the percentage of correctly identified vulnerabilities strictly against the vulnerabilities intentionally implemented into the systems. While this list is numerically limited, it is not arbitrary; it was purposefully designed as a representative Ground Truth sample covering diverse attack vectors, such as non-standard ports, outdated software, and unencrypted protocols. It is defined as the ratio of true positives (TP) to the total number of explicitly implemented vulnerabilities (C).

$$WS = \frac{TP}{C} \cdot 100\%$$

- Time efficiency (T_{scan}) measures the total duration of the scanning process.
- Resource utilization (WZ) evaluates utilization of machine hardware, focusing on the central processing unit and random access memory. To calculate resource utilization arithmetic mean of the average CPU load ($\bar{U}_{CPU}$) and the percentage of utilized RAM ($\bar{U}_{RAM}$ relative to RAM_{total}) is used.

$$WZ = \frac{\bar{U}_{CPU} + \left(\frac{\bar{U}_{RAM}}{RAM_{total}} \cdot 100\% \right)}{2}$$

- Usability (WU) is a metric that assesses the ergonomic aspects of scanning tools. It involves a configuration process (P_k), user interface (P_i) and reporting mechanisms (P_r). Each feature is scored on a one-to-three scale.

$$WU = P_k + P_i + P_r$$

Additionally, two more metrics were formed:

- Detection volume (L) is the count of vulnerabilities identified during a testing scenario.
- U/B ratio (R) is a metric used to check the impact of authenticated scanning. It represents the ratio of vulnerabilities detected in the authenticated scenario (U) to those found in the non-authenticated scenario (B).

$$R = \frac{U}{B}$$

6.2. Multi-criteria analysis using the SAW method

The final evaluation of network vulnerability scanners requires adding up different values and factors. To select the best software, a multi-criteria decision-making approach based on simple additive weighting was used. This method allows adding previously mentioned metrics into a single assessment. Because the evaluated parameters possess different units, these must be normalized to a common scale from zero to one. Metrics were also divided into two main categories:

- Beneficial criteria are parameters where a higher value indicates better performance, for example, targeted detection rate or usability.
- Non-beneficial criteria are categories where a lower value indicates better performance, for example, time efficiency or resource utilization.

The final evaluation score for each scanner is calculated as the sum of the products of normalized criteria values and their assigned weights, according to the following formula:

$$O = \sum_{j=1}^n W_j \cdot X_{ij}$$

Where O is the final score of the specific scanner, W_j is the weight determining the importance of a given criterion, X_{ij} is the normalized value of the j -th criterion for the i -th scanner and n is the total number of analyzed criteria.

6.3. Expert weight allocation

To ensure reliable assessment, each of the four criteria was assigned a weight (W_j) using an expert method. The weight represents real-world priorities regarding security audits. Weights are defined as follows:

- Targeted Detection Rate ($W_1 = 0.60$) is the most important criterion. The main purpose of a vulnerability scanner is to detect security holes in computer environments; therefore, this parameter is the most critical one.
- Usability ($W_2 = 0.20$) is the second most important factor. From an engineer's point of view, an ergonomic and user-friendly environment allows him to do his job faster and without making mistakes.
- Time efficiency ($W_3 = 0.10$) is the third most important factor. It is crucial when dealing with potentially critical vulnerabilities or when conducting as-

assessments in production environments constrained by limited maintenance windows.

- The last one is resource utilization ($W_4 = 0.10$) with the lowest weight. Modern workstations, servers, and network infrastructure generally possess high computing performance, making the hardware load generated by the tool the least crucial.

6.4. Scenario A: Network identification of attack surface

The primary objective of this scenario is to evaluate the time efficiency and accuracy of the tools in mapping the network topology, identifying active hosts, and enumerating running services. This phase simulates a standard reconnaissance procedure utilized by security engineers to rapidly profile newly introduced network assets and determine the initial attack surface. To ensure methodological consistency, all tools were restricted to basic identification tasks without executing intrusive vulnerability payloads. Nmap was configured to perform TCP SYN stealth scanning, service version enumeration, and operating system fingerprinting. Correspondingly, OpenVAS was executed utilizing its dedicated discovery profile, and Nessus was configured with its baseline host discovery policy. This approach ensured that all scanners operated under equivalent conditions.

6.5. Scenario B: Scanning vulnerabilities without authentication

The objective of this scenario is to evaluate the tool's performance in an unauthorized environment. This approach simulates a realistic threat model where an unauthorized person attempts to identify and exploit system vulnerabilities to breach into a system. Nmap was utilized with its Nmap Scripting Engine configured for vulnerability detection. OpenVAS was executed utilizing its comprehensive "Full and fast" unauthenticated profile, while Nessus operated under its standard, unauthenticated network assessment policy. This methodology allowed for a direct comparison of the tools detection rate without providing them with privileged access to target systems.

6.6. Scenario C: Vulnerability audit using an authentication method

The primary objective of this scenario is to conduct a vulnerability audit utilizing authenticated scanning. In this configuration, the scanners are provisioned with administrative credentials, allowing them to bypass external firewalls, access local registries, and perform deep patch-level auditing of the installed software. Nmap, being fundamentally designed for network reconnaissance and lacking a native credentialed patch management architecture, was excluded from this specific scenario. OpenVAS and Nessus were executed utilizing their most comprehensive authenticated profiles to ensure maximum detection accuracy. The empirical data gathered in this scenario is critical for calculating the U/B Ratio, which quantifies the variance

in detection efficacy between authenticated and unauthenticated assessments.

7. Results and discussion

This chapter presents results obtained during the execution of scenarios. With the methodology established before, three tools were used. The metrics described in the Methodology section were monitored and recorded.

7.1. Attack surface discovery (Scenario A)

The primary objective of Scenario A was to evaluate the scanner's capabilities for host detection within the subnet, the accuracy of determining the operating systems, and the precision of enumerating open TCP ports and their underlying services. A critical parameter evaluated during this scenario was detection precision, which was specifically tested against a non-standard HTTP service intentionally deployed on port 8888 within the Windows Server. The aggregated results obtained during the discovery scenario are presented in Table 4.

Table 4: Summary of scenario A results

Parameter	Nmap	OpenVAS	Nessus
Detected active devices	6	6	6
Scanning time (T_{scan}) in seconds	247	9120	360
OS ID precision	High	Low	Medium
Hidden HTTP service (port 8888)	Not detected	Detected	Detected
Average CPU load ($\bar{U}_{CPU}$)	3.90%	4.66%	18.35%
Average RAM usage ($\bar{U}_{RAM}$)	2.51 GB	2.33 GB	2.22 GB

An in-depth analysis of the metrics obtained reveals significant architectural differences between the tested solutions. All three tools successfully identified the complete pool of active host devices. However, Nmap showed the best time efficiency, concluding the scan in just 247 seconds with a minimal CPU load. This shows that Nmap is highly optimized for rapid network reconnaissance. Furthermore, Nmap was most precise in operating system identification with its accurate versions of OS results.

OpenVAS took 9120 seconds to finish a basic discovery scan. This indicates that its discovery profile is heavily unoptimized.

In terms of found ports, Nessus outperformed the other two by identifying 70 open TCP ports, compared to Nmap (37) and OpenVAS (33). There was a clear difference in how well the hidden HTTP service on port 8888 was detected. OpenVAS and Nessus successfully detected this non-standard service. Nmap failed to identify it during this scenario, which is a direct consequence of its default setup that scans only the top 1000 most common TCP ports, bypassing port 8888. However, Nessus demonstrated a slight deficiency in operating system identification, specifically failing to accurately identify the underlying system of the Metasploitable 2 Linux machine.

7.2. Unauthenticated vulnerability scanning (Scenario B)

This scenario evaluated the effectiveness of the scanners in an unauthenticated mode, simulating a threat actor without knowledge about the systems. In this scenario, the tools searched the network for vulnerabilities and classified the identified threats using the CVSS (Common Vulnerability Scoring System). The aggregated results gathered during the scenario are presented in Table 5.

Table 5: Summary of scenario B results

Parameter	Nmap	OpenVAS	Nessus
Scanning time (T_{scan}) in seconds	757	4020	2880
Detected vulnerabilities (L)	10	70	62
Identified CVE	336	108	39
CVSS threat classification	None	Critical: 14, High: 8, Medium: 41, Low: 7, Info: 153	Critical: 10, High: 9, Medium: 33, Low: 10, Info: 376
Average RAM usage ($\bar{U}_{RAM}$)	2.18GB	2.54 GB	2.25 GB
Average CPU load ($\bar{U}_{CPU}$)	3.35%	30.56%	11.33%

Table 6: Results of targeted services in scenario B

Application / Service (Port)	Nmap	OpenVAS	Nessus
Vsftpd (21)	Detected	Detected	Detected
UnrealIRCd (6667)	Detected	Not detected	Detected
Java RMI (1099)	Not detected	Not detected	Not detected
Telnet (23)	Detected	Detected	Detected
Mozilla Firefox	Not detected	Not detected	Not detected
VLC Media Player	Not detected	Not detected	Not detected
7-Zip	Not detected	Not detected	Not detected
Weak system user credentials	Not detected	Detected	Detected
http (8888)	Detected	Detected	Detected
SMB	Not detected	Detected	Not detected
FTP, Telnet	Detected	Detected	Detected

To further evaluate detection accuracy, the results were compared against the specific vulnerabilities implemented in the test environment. The result for these services is shown in Table 6.

These results show clear differences in how each tool approaches vulnerability scanning. Nmap was the fastest tool, with 757 seconds in time. It also performed best when it came to resource consumption. Generating only a 3.35% CPU load. However, it reported only 10 distinct vulnerabilities but linked them to 336 different CVEs. This happened because Nmap NSE lists all possible CVEs that are associated with a detected service. Furthermore, Nmap does not provide a CVSS risk classification, which makes it harder to prioritize threats.

Among dedicated scanners, OpenVAS detected the highest number of vulnerabilities, which are 70 in total, and the highest number of CVEs, which is 108. However, this came at a high cost of time and CPU load. Full

OpenVAS scan took 4020 seconds in total, and the average processor load was 30.56%. Nessus offered a better balance between time and performance, finishing the scan in 2880 seconds with moderate CPU usage at 11.33%, while finding 62 vulnerabilities.

When analyzing the targeted vulnerabilities (see Table 6), an observation can be made regarding local applications (Mozilla Firefox, VLC Media Player, 7-zip). None of the scanners was able to detect these outdated versions of programs.

7.3. Authenticated vulnerability scanning (Scenario C)

This scenario was focused on conducting a scan with administrator privileges. Providing the scanners with credentials allowed to test if they can perform a deep, local security audit. As defined in the methodology, Nmap was excluded from this scenario due to its lack of native credentialed scanning architecture. The aggregated performance and detection metrics for OpenVAS and Nessus are presented in Table 7.

Table 7: Summary of scenario C results

Parameter	OpenVAS	Nessus
Scanning time (T_{scan}) in seconds	3780	3120
Detected vulnerabilities (L)	70	301
Identified CVE	111	837
CVSS threat classification	Critical: 14, High: 8, Medium: 41, Low: 7, Info: 191	Critical: 32, High: 98, Medium: 150, Low: 21, Info: 448
Average RAM usage ($\bar{U}_{RAM}$)	2.74 GB	2.38 GB
Average CPU load ($\bar{U}_{CPU}$)	26.51%	20.99%

To verify the precision of the credentialed scans, the results were matched against the specific vulnerabilities implemented in the test environment. The result for this test is shown in Table 8.

Table 8: Results of targeted services in scenario C

Application / Service (Port)	OpenVAS	Nessus
Vsftpd (21)	Detected	Detected
UnrealIRCd (6667)	Detected	Detected
Java RMI (1099)	Not detected	Not detected
Telnet (23)	Detected	Detected
Mozilla Firefox	Not detected	Not detected
VLC Media Player	Not detected	Not detected
7-Zip	Not detected	Not detected
Weak system user credentials	Detected	Detected
http (8888)	Detected	Detected
SMB	Not detected	Not detected
FTP, Telnet	Detected	Detected

Implementation of administrative credentials revealed a difference in the effectiveness of the tested tools, which can be shown using the U/B ratio (R). Nessus performed very well in these tests. Its detection volume (L) spiked from 62 in scenario B to 301 in scenario C, resulting in U/B ratio of $R \approx 4.85$. This proves that Nessus successfully utilized the provided credentials to log into systems and do an internal audit, discovering 836 specific CVEs.

On the other hand, OpenVAS detection volume remained at 70 vulnerabilities. This was an identical value as in scenario B. Furthermore, OpenVAS was less efficient when it comes to computing resources, requiring more time (3780 seconds) and generating a higher CPU load (26.51%) than Nessus.

Another important finding is that neither scanner detected the old local application versions, as shown in Table 8. This highlights a basic requirement for authenticated Windows scanning to work. The remote registry service needs to be enabled because, without that, scanners are unable to remotely verify installed software versions. This demonstrates that administrative access alone is insufficient for a comprehensive audit.

7.4. Multi-criterial evaluation results

To combine the results and find the most effective scanner, the SAW algorithm was applied to the collected data. The decision matrix uses data from the scan results. Usability was rated using setup, interface, and reporting criteria. Technical metrics were gathered directly from scanning data.

Criteria were classified as beneficial or non-beneficial. Table 9 presents an unnormalized decision matrix.

Table 9: Raw data decision matrix for scanners

Criterion	Type	Weight (W_j)	Nmap	Open-VAS	Nessus
Targeted Detection Rate (WS)	Max	0.60	45.45	54.54	54.54
Usability (WU)	Max	0.20	5	6	8
Time efficiency (T_{scan})	Min	0.10	757	3900	3000
Resource utilization (WZ)	Min	0.10	14.98	30.38	22.21

Following the SAW methodology, these raw values were mathematically normalized to a common scale (0-1). For beneficial criteria, the value was divided by the maximum value in the dataset. For non-beneficial criteria, the minimum value in the dataset was divided by the given value. The final evaluation score was calculated by summing the normalized values and their weights. The normalized matrix and the final rankings are detailed in Table 10.

Table 10: Normalized decision matrix and final score

Normalized criterion	Nmap	OpenVAS	Nessus
Targeted Detection rate (WS)	0.83	1	1
Usability (WU)	0.63	0.75	1
Time efficiency (T_{scan})	1	0.19	0.25
Resource utilization (WZ)	1	0.49	0.67
Final score (O)	0.825	0.819	0.893

The SAW method results clearly show that Nessus is the best vulnerability scanner in this comparison, achieving the highest final score $O = 0.893$. Nessus won mostly because it performed best in the two most important categories, which are detection rate and usability. Even though Nmap could not perform authenticated scans, it still scored slightly higher than Open-

VAS. OpenVAS successfully found the same number of target vulnerabilities as Nessus, however, it lost a lot of points because it was very slow and used too much CPU. On the other hand, Nmap made up for its lower detection rate by being extremely fast and lightweight, getting maximum points in both time efficiency and resource utilization.

7.5. Dataset limitations

While the configured environment provided a stable baseline for the multi-criteria evaluation, the constructed dataset has specific limitations. The testbed was restricted to three operating systems: Windows Server 2019, Windows 11, and Metasploitable 2, and explicitly tracked a limited pool of targeted vulnerabilities. Although this sample effectively represents diverse attack vectors such as known backdoors, outdated local software, and unpatched protocols, it does not reflect the complexity and massive scale of enterprise networks. Furthermore, the dataset does not evaluate the scanners' performance against zero-day exploits or cloud-native environments. As a result, the findings are strictly applicable to traditional, on-premise network architectures. Acknowledging these limitations ensures that the calculated targeted detection rate is interpreted as a controlled benchmark rather than an absolute global efficacy metric. Moreover, the pool of intentionally implemented vulnerabilities is highly concise and restricted to a strictly defined subset of known CVEs. Consequently, while it accurately measures detection precision for these exact vectors, it cannot fully represent the scanners' effectiveness against a broader, unrestricted range of security flaws.

7.6. Discussion

The obtained results reveal that the performance metrics of the evaluated scanners are strictly tied to their underlying architectural philosophies. Nmap's failure to detect the intentionally hidden HTTP service and its overall low detection volume are direct consequences of its design as a rapid network reconnaissance tool. By default, its engine prioritizes scanning speed by probing only a predefined subset of the most common ports. Furthermore, its reliance on NSE scripts makes it fundamentally unsuitable for deep, local system audits compared to dedicated scanners. Conversely, the extreme resource consumption and prolonged execution times observed with OpenVAS highlight a significant optimization gap typical of some open-source solutions. While OpenVAS successfully matched commercial software in identifying the core network vulnerabilities, its processing engine executes Network Vulnerability Tests with substantial computational overhead, severely limiting its scalability in large environments. The most critical architectural inequality was exposed during the authenticated scanning phase. Nessus's detection volume spiked dramatically after logging into the systems, proving the high efficiency of its commercially maintained plugin database. OpenVAS's failure to increase its detection rate post-authentication suggests an inabil-

ity of its engine to properly utilize the provided Windows credentials under the configured test conditions. Furthermore, the inability of both dedicated scanners to detect intentionally outdated local applications effectively demonstrates a fundamental rule of vulnerability management: administrative credentials are entirely insufficient if the target operating system actively restricts remote auditing protocols, such as a disabled remote registry service.

8. Conclusions

The primary thesis of this paper states that effective vulnerability detection requires using various tools with different functions, as no single tool can identify all security threats. The test results and the SAW algorithm evaluation fully confirmed this thesis.

The study successfully verified all the formulated hypotheses. First, it confirmed that dedicated automated scanners, such as Nessus and OpenVAS, discover significantly more vulnerabilities than Nmap. Despite that, obtaining a complete security picture requires combining multiple tools. Nmap is extremely fast and lightweight, making it ideal for quick network discovery, while Nessus and OpenVAS are necessary for deep system analysis. Furthermore, the evaluation proved that execution time and resource usage depend heavily on the chosen tool and the depth of the analysis. Nmap was the fastest because it only scans the network surface. In contrast, OpenVAS and Nessus took much longer to perform their deep checks, with OpenVAS being the slowest and using the most CPU resources. The research also showed that authenticated scanning drastically increases the detection rate compared to unauthenticated scanning.

The research showed that free tools like OpenVAS can detect the same specific target vulnerabilities as commercial tools like Nessus. However, OpenVAS suffers from poor optimization. It consumes significantly more time and processing power. It also lacks constant support from the developer's side.

The multi-criteria SAW decision model conclusively identified Nessus as the best overall vulnerability scanner with a final score of 0.893. It offers the best balance between high detection rate, ease of use, and good resource optimization, making it the most reliable choice. In conclusion, the methodology applied in this study successfully created a clear and objective framework for comparing security tools.

References

- [1] S. Mitchell, Mandiant report finds rise in financially motivated cyber attacks, <https://securitybrief.asia/story/mandiant-report-finds-rise-in-financially-motivated-cyber-attacks>, [11.11.2025].
- [2] N. Mihaylov, Cyber Dimensions of a Hybrid Warfare, <https://cyberpeaceinstitute.org/news/cyber-dimensions-of-a-hybrid-warfare/>, [11.11.2025].
- [3] Help Net Security, Edgescan 2025 Vulnerability Statistics Report, <https://www.helpnetsecurity.com/2025/04/30/edgescan-2025-vulnerability-statistics/>, [02.02.2026].
- [4] S. Toyin, Comparative analysis of security vulnerability scanners (Nessus and OpenVAS) in cloud environment, Master thesis, University of Bolton, Bolton 2023.
- [5] J. Samera, M. Mapfirakupa, Comparative Analysis of Network Scanning Tools for Windows and Ubuntu VMs, *International Journal of Engineering Management and Humanities* 6(2) (2025) 29-38, <https://doi.org/10.13140/RG.2.2.23826.98242>.
- [6] I. Chalvatzis, D. Karras, R. Papademetriou, Reproducible modelling and simulating security vulnerability scanners evaluation framework towards risk management assessment of small and medium enterprises business networks, *Indian Journal of Science and Technology* 13(37) (2020) 3910-3943, <https://doi.org/10.17485/IJST/v13i37.868>.
- [7] F. Alghamdi, A Comparative Analysis of Network Enumeration Tools, *Journal of Computer Science & Computational Mathematics* 14(4) (2024) 21-26, <https://doi.org/10.20967/jscsm.2024.04.001>.
- [8] A. Chaturvedi, B. Lakhani, T. Agarwal, Mohana, M. Moharir, A. Kumar, A Comprehensive Vulnerability Tools Analysis for Security and Control in IT Environment and Organizations, In 2024 5th International Conference on Electronics and Sustainable Communication Systems (ICESC) IEEE (2024) 612-618, <https://doi.org/10.1109/ICESC60852.2024.10689860>.
- [9] Accio, 2025 Operating System Market Share Trends: Key Insights & Predictions, <https://www.accio.com/business/operating-system-market-share-trend>, [25.01.2026].
- [10] R. Sharma, R. Sibal, S. Sabharwal, Change Point Modelling in the Vulnerability Discovery Process, In: A. Luhach, D. Singh, P. Hsiung, K. Hawari, P. Lingras, P. Singh, (eds) *Advanced Informatics for Computing Research. Communications in Computer and Information Science*, Springer 956 (2019) 559-568, https://doi.org/10.1007/978-981-13-3143-5_46.