UNIVERSITÄT ZU LÜBECK
INSTITUT FÜR IT-SICHERHEIT

Comparing vulnerability management to automated penetration testing

Vergleich von Schwachstellenmanagement und automatisierten Penetrationstests

Bachelorarbeit

im Rahmen des Studiengangs
IT-Sicherheit
der Universität zu Lübeck

vorgelegt von
Felix Zenk

ausgegeben und betreut von
Prof. Dr. Thomas Eisenbarth

Die Arbeit ist im Rahmen einer Tätigkeit bei der Firma Axians IT Security GmbH entstanden.
This thesis was created within an activity at the company Axians IT Security GmbH.

Lübeck, December 17, 2024

Abstract

In today's world cyberattacks are ubiquitous and dangerous to the point of threatening an organization's existence. To avert the dangers of cybercrime it is essential to develop a good cybersecurity strategy. Past work has shown that not only the completeness of the vulnerabilities found but also their prioritization in remediation plays a decisive role. This bachelor thesis examines the concepts of "vulnerability management" and "automated penetration testing" to establish a guideline that when considering the growth of an organization's network, yields which of the two concepts is more suitable for that organization's current situation. Vulnerability management describes the process of passively scanning networks for indications of vulnerable software running on hosts. Automated penetration testing is an active approach, that tries to perform a real attack on the target to prove the existence of a vulnerability. Three test networks with a growing number of hosts and variable amounts of exploitable services were built and scanned with a selection of commercial and FOSS vulnerability scanners to obtain a broad overall picture, that compares both concepts. The prioritization of a vulnerability is realized with the CVSSv3 score in the commercial tools, the FOSS alternatives rely on the older CVSSv2 standard. Finding viable FOSS alternatives has proven to be the most challenging part of this work, as fully automated open-source penetration testing tools are sparse. The scan results show that an organization should start by using automated penetration testing and later split their focus into server and client systems. After a while, both approaches converge in detection rate for server systems, whereas vulnerability management proved dominant for clients.

Zusammenfassung

Cyberangriffe sind in der heutigen Welt allgegenwärtig und so gefährlich, dass sie die Existenz einer Organisation bedrohen können. Um die Gefahren der Cyberkriminalität abzuwenden, ist es unerlässlich, eine gute Cybersicherheitsstrategie zu entwickeln. Frühere Arbeiten haben gezeigt, dass nicht nur die Vollständigkeit der gefundenen Schwachstellen, sondern auch deren Priorisierung bei der Behebung eine entscheidende Rolle spielen. Diese Bachelorarbeit untersucht die Konzepte "Schwachstellenmanagement" und "automatisierte Penetrationstests", um eine Richtlinie zu erstellen, die unter Berücksichtigung des Wachstums des Netzwerks einer Organisation darstellt, welches der beiden Konzepte für die aktuelle Situation dieser Organisation besser geeignet ist. Schwachstellenmanagement beschreibt den Prozess des passiven Scannens von Netzwerken auf Hinweise über anfällige Software, die auf Hosts ausgeführt wird. Automatisierte Penetrationstests sind ein aktiver Ansatz, in dem versucht wird, einen echten Angriff auf das Ziel durchzuführen, um die Existenz einer Schwachstelle zu beweisen. Drei Testnetzwerke mit einer wachsenden Anzahl von Hosts und unterschiedlichen Mengen an ausnutzbaren Diensten wurden aufgesetzt und mit einer Auswahl an kommerziellen und FOSS-Schwachstellenscannern gescannt, um ein breites Gesamtbild zu erhalten, das beide Konzepte vergleicht. Die Priorisierung einer Schwachstelle wird in den kommerziellen Tools mit dem CVSSv3-Score realisiert, die FOSS-Alternativen basieren auf dem älteren CVSSv2-Standard. Die Suche nach brauchbaren FOSS-Alternativen erwies sich als der schwierigste Teil dieser Arbeit, da es nur wenige vollautomatische Open-Source-Penetrationstest-Tools gibt. Die Scan-Ergebnisse zeigen, dass eine Organisation mit automatisierten Penetrationstests beginnen und seinen Fokus später auf Server- und Client-Systeme aufteilen sollte. Nach einiger Zeit nähern sich beide Ansätze hinsichtlich der Erkennungsrate für Server-Systeme an, während sich das Schwachstellenmanagement für Clients als dominant erwies.

Erklärung

Ich versichere an Eides statt, die vorliegende Arbeit selbstständig und nur unter Benutzung der angegebenen Hilfsmittel angefertigt zu haben.

Lübeck, December 17, 2024

Contents

1 Introduction

Today, many different organizations are increasingly threatened by cyberattacks [LTT+23]. Even though cybercrime in Germany has been declining over the last years foreign cybercrime is booming. The number of recorded cyber crimes has doubled in just four years since the year 2020 [BKA24]. Massive amounts of damage can be caused to an organization by just a single cyberattack.
Globally, more than one in four recorded crimes are cyber crimes [BKA24]. The most prevalent type of cybercrime nowadays is ransomware [BSI23, LTT+23]. More and more victims of ransomware attacks refuse to pay the ransom, which forces cybercriminals to increase the amount of the ransom per attack. The average ransom demand for a single attack was $621,858 in 2023 and total extorted ransoms reached over $1B for the first time [BKA24]. A single attack is however not the only concern of victims, as it is now common practice amongst ransomware groups to perform a triple extortion technique by first encrypting files and demanding a ransom from the victim organization, then threatening to publicize transferred data and, as a third step, blackmailing the customers of that victim and also demanding a ransom from them [BKA24]. In 2022 the amount of damage inflicted on organizations because of cybercrime has quadrupled over the last 5 years before that and is now quantified at $202.7B in total damages per year across german organizations [BKA23]. All the above contributes to small and medium-sized organizations (SMOs) being existentially threatened by just a single cyberattack [BKA24, BSI23]. And the bad news is, that ransomware as a service (RaaS) groups get increasingly professional in their scheming. Cybercrime groups are getting structured similar to legal companies with different departments and a clear hierarchy. The result is, that RaaS is becoming vastly available and cheaper, so more organizations become targets of criminals. 59% of all ransomware victims in Germany are medium-sized organizations and a further 9% resemble small organizations and large corporations [BSI23]. It is apparent that organizations starting their business or expanding are especially at risk of cybercrime, therefore an efficient and working cybersecurity strategy is important and could save organizations from bankruptcy. Most SMOs do not recognize this danger, as 54% of them do not get outside help from a managed security service provider (MSSP), but instead follow the directive “We’ll manage it ourselves somehow.” [BSI23] leading to these alarming numbers. In 2022 a total of $7.8B was invested by german organizations towards cybersecurity. In comparison to the $202.7B in losses due to the lack of cybersecurity measures, it is highly

lucrative for organizations to invest in their cybersecurity.
In order to protect themselves against these attacks, it is important for organizations to recognize and rectify vulnerabilities in their own systems and fix them. *Vulnerability management* and *penetration testing* are two widely used concepts that aim to uncover vulnerabilities in a system, but differ in their approach [DT15]. Both concepts can be carried out as an internal or external approach. Internal scans are performed from within the organization network and can thus provide deeper insights into the system. In addition, credentials can be used to access systems and thus also find vulnerabilities that are only visible from the perspective of a logged-in user. An external scan is carried out from outside the organization network and can only find externally visible vulnerabilities (attack surface) [MK15].

1.1 Background

Penetration testing has the potential to penetrate deeper into a system and uncover more vulnerabilities, because vulnerabilities are actively exploited and therefore can be verified to actually work. These vulnerabilities can then be further used to carry out an attack chain consisting of multiple exploits to eventually compromise a system.
Vulnerability management, on the other hand, can cover a larger area as it does not only focus on exploitable vulnerabilities, but also on vulnerabilities that are not relevant for an attack at this time, but are nevertheless present and can become relevant in the future. It does not actively exploit vulnerabilities, but instead takes a passive approach. Vulnerability management works by scanning the system for information used for service detection and revealing version information of these services. The collected information is then cross-referenced with vulnerability databases to identify vulnerable versions of software running on that system. This is the reason why vulnerability management can ever only find the uppermost layer of vulnerabilities in a system, if no credentials were provided for a service enumeration on the system itself. In addition, vulnerability management is generally easier to use and, for the same systems with credentials provided, delivers similarly good results more quickly [Sha20].
Penetration testing comes in two different forms: Automated penetration testing and manual penetration testing. Automated penetration testing refers to the process in which a tool automatically attempts to penetrate a system. Various attack methods are tried out in the process, to find vulnerabilities. These vulnerabilities can be combined to form an attack chain and thus enable the deepest possible penetration into the system. Automated penetration testing is a relatively new concept, the traditional approach is a manual penetration test. Manual penetration testing differs from automated penetration test-

ing in that a human carries out the attacks manually and not an automated tool. Comparisons between automated penetration testing and manual penetration testing have shown that automated penetration testing is generally faster and more efficient and at the same time does not find significantly fewer vulnerabilities than a manual test would find [ADA18, Sha20].

Only some scenarios are still too complex for automated tests and require human intervention, for example custom captchas or multifactor authentication [Sha20]. With the wide spread of AI technology and knowledge it is possible however to design specialized models that are able to defeat captchas [DH23]. Furthermore, automated penetration testing is generally more economical than a manual test, as it is more cost-effective and requires less human working time and can be carried out independent of the time of day [ADA18]. Human error is also avoided in the process as the tests run automated without human intervention [ADA18]. One advantage of manual penetration testing, on the other hand, is that it is more flexible and can be adapted to specific requirements and the context of a network. It is important to avoid false positives which automated penetration testing ensures by testing whether exploits actually work. This keeps the false positive rate of automated penetration testing lower than the one of vulnerability management [ADA18, Sha20]. A particular advantage of manual tests is that they can discover zero-day exploits that are not yet listed in the exploit databases or strategies of automated tools [Sha20].

In summary, organizations that want cost-efficient and effective protection against cyber-attacks, should consider the approach of automated penetration testing as opposed to manual penetration testing. When looking at the completeness of the results of both approaches, it can be seen that both approaches are able to find most vulnerabilities in a system, only zero-day exploits and very specific attacks require manual testing [ADA18]. This may change over time, as automated penetration testing was effectively developed in the past 10 years and likely will see further improvements in the future. In general, it can be said that automated penetration testing and vulnerability management are likely complementary approaches which, in combination, provide a complete and comprehensive picture of the vulnerabilities in a system [DT15, Sha20].

Vulnerability management, on the other hand, refers to the process in which a system is passively examined for vulnerabilities. A system's services are probed and their version identified and then compared to a database of vulnerable versions of software. In particular, port scanners (most of the time nmap) are used, which establish connections on a list of defined ports of a system in order to test whether they are open and offer a service. Possible services running on the host are inferred by the port number, if they run on standardized ports such as an HTTPS web server on TCP/443, or probed to grab banners or a handshake sequence that can identify the service. This initial test is followed by additional tests that determine more details about the system such as operating system or service version. The findings of this scanning phase is then checked against vulnerability databases, such as the National Vulnerability Database (NVD) [BRW13], the GitHub Advisory Database (GHAD) [GHA24] or the Common Vulnerabilities and Exposures (CVE) [CVE] list. Matches are reported as existing vulnerabilities, without any attempt to actively exploit these vulnerabilities.

This is known as a VAPT (vulnerability assessment and penetration testing) approach. [SM15] confirmed in their case study, where they found four additional vulnerabilities in a bank's web applications, that manual penetration testing is a good supplement to vulnerability management. [KKSG19] presents a similar evaluation of the VAPT process, where they come to the conclusion that implementing a working VAPT process is crucial for organizations today and in the future. The question remains whether automated penetration testing can compete with manual penetration in this regard and when not looking at just web application vulnerabilities, but the network as a whole. Past work has shown, that the comparatively high false-positive rate of vulnerability management can be reduced by automated penetration testing, as this only finds vulnerabilities that can be actively exploited. Automated penetration testing can thus provide an overview of the acute vulnerabilities in a system, while vulnerability management can provide an overview of all vulnerabilities in a system, even if they cannot be actively exploited and are not relevant at the time, but may become relevant in the future. Such vulnerabilities, that are not found by automated penetration testing because they do not result in a promising attack vector can still be found and remedied. This creates a comprehensive picture of the vulnerabilities in a system, which makes it possible to prioritize the vulnerabilities according to their severity and thus fix the most critical vulnerabilities first [Sha20]. This is particularly important as past work has shown that it is not only the completeness of the vulnerabilities that is important, but also the prioritization of said vulnerability plays a relevant role for the remediation [WOS21]. Automated penetration testing therefore has the promise to do the same as manual penetration testing and aid vulnerability management in finding a complete yet prioritized picture of an organizations vulnerabilities.

Various scoring systems, such as the CVSS [CVS15] or manufacturer-specific scoring systems, can help to assess vulnerabilities according to their severity [SSN23, WOS21]. The CVSS score is a widely used scoring system that rates vulnerabilities according to their severity. It consists of several metrics, such as the severity of the impact, the probability of a successful attack and the availability of exploits [WOS21]. The score is available in different versions, with the latest version being the recently published CVSSv4 score. Tenable, a leading provider of vulnerability management tools, builds on the CVSSv3 score by taking other metrics, such as exploit availability or the current occurrence of attacks into account to build their own Vulnerability Priority Rating (VPR) score in an attempt to address the issue of unclear remediation prioritization of the CVSSv3. The new CVSSv4 score also includes the system's environment, however, as it has only been in existence for about a year, it is not yet as widespread as the CVSSv3.

1.2 Scope of this work

The challenge for organizations is to find the right balance between the two approaches in order to develop an effective and efficient cybersecurity strategy. It is often difficult to decide which approach should be pursued first in order to ensure the greatest possible security. This work tries to answer this question by comparing both concepts in growing network environments.

Both concepts could also be compared regarding the organization's cybersecurity maturity level [WOS21, Mie20]. There are many guidelines like the ISO-27000 series and the NIST SP 800 series that lay out rules to follow for better cybersecurity. Many cybersecurity maturity frameworks implement those guidelines in their models. The Capability Maturity Model Integration (CMMI) [Yam17] by ISACA, the NIST Cybersecurity Framework 2.0 [CSF24], the CIS Community Defense Model 2.0 [Sto21] from the Center for Internet Security and the Gartner Score [aia19] are well-known cybersecurity maturity models. Compliance with the requirements of such frameworks can be either self assessed or certified depending on the framework. Those models not only consider the hardware, software and configuration of networks, but also the organizational environment including the people and processes. Therefore, evaluating results of this work according to cybersecurity models is not applicable, because simulating an entire organization would be too much for this work.

Furthermore, specialized maturity models exist. The ENISA CSIRT framework [fNS19] for example is a model for CSIRTs (Cybersecurity incident and response team) that applies only to a specialized unit within the organization or an external contractor. In Germany, the ISO 27001 standard [ISO05] is a widely used framework, whereas the NIST Cybersecurity Framework [CSF24] is the most widely used worldwide.

1.3 Contributions

In this thesis, the concepts of "automated penetration testing" and "vulnerability management" are examined in terms of vulnerabilities found in each case. The aim is to establish a guideline or recognizable trend for decision-making that, taking network growth into account, should show which of the two concepts is better suited to the current situation of an organization.

A test network was set up that contains vulnerable systems to test the scanners against each other. Vulnerability scans were carried out with both commercial and open source tools, to create the broadest possible picture that compares the concepts. The commercial tools used are Pentera Core [Pen15] from Pentera for automated penetration testing and the Tenable Nessus Professional [Der98] scanner from Tenable for vulnerability management.
The open source alternatives used are the jok3r project [jok17] for automated penetration testing and Greenbone Community Edition [Gre07] with the OpenVAS for vulnerability management.
Vulnerability management and automated penetration testing are performed in the internal variant, as the results should be as comprehensive and meaningful as possible for the organization and not only find superficial vulnerabilities from an attacker's perspective. No credentials for scanning installed software were provided.
In particular, this work has the following objectives:

1. Compare the concepts of "vulnerability management" and "automated penetration testing" in terms of vulnerabilities found by each approach.

2. Establish a guideline or trend to follow for decision-making in consideration of the network growth that shows what concept is better suited for the current situation of the organization.

1.4 Axians IT Security GmbH

This bachelor thesis was written in cooperation with Axians IT Security GmbH (AITSec). The Axians Group is part of the global brand network for ICT solutions from VINCI Energies. AITSec is the organization of the group specializing in cybersecurity and acts as a managed service provider that carries out vulnerability scans and automated penetration tests for customers at regular intervals. The lab environment for conducting scans, including the resources for virtual machines and licenses for commercial products were provided by AITSec for this work. The VLAN and firewall policies for the lab environment were configured by AITSec.

1.5 Organization of this thesis

This thesis is organised as follows:
Chapter 2 looks into previous work, first considering similar research, then more on the theoretical background of the concepts. Followed by an outlook on the relevance of this work and research comparing tools and methods used at the end.

The Chapter 3 starts by describing the modeled situations. After that, the specific test environments for each network are listed. Then, the tools used for scanning the networks and the process of finding suitable open source tools for this work is documented. The chapter closes with the configurations for each scanner that were used in the scans.
Chapter 4 presents the results data and interpretations of the results. The chapter begins with the evaluation of the scan reports that form the resulting data. A simple overview on the numbers with obvious observations is given, further processed and examined in the context of comparing the two approaches. Thereafter, the limitations that presented themselves during the work on the project are laid out. Lastly, interpretations of the results that answer the hypotheses are given.
Finally, in Chapter 5 the most important findings of this thesis are concluded and open problems that provide further research directions are discussed.

2 Related work

This section is split into: Similar work, theoretical background on the concepts, relevance of this work, methodologies and tools used in the field of cybersecurity.

2.1 Similar work

Mietala [Mie20] investigated when an organization should start with vulnerability management. Both internal and external scans were considered. The work mainly examines various cybersecurity frameworks and maturity models to determine when an organization should start using vulnerability management. In addition, various commercial vulnerability management scanners are presented. The work mentions penetration testing as a possible way to find vulnerabilities, but does not go into more detail about penetration testing. The researcher found out, that the Sans institute guide for implementing a vulnerability management process and CRR Supplemental Resource Guide for vulnerability management provide an easily adaptable vulnerability management implementation process. Cybersecurity frameworks can be overwhelming for small companies, but these guidelines can theoretically be followed by them. Cybersecurity maturity models on the other hand are best utilized by medium-sized or larger enterprises, that can invest the time and effort to follow all controls set by the model. The Gartner IT Score provides the easiest entry in this regard. A specific point in time for when an organization should start vulnerability management could not be determined however and should instead be estimated by the organization itself.

2.2 Theoretical background

Doshi et al. [DT15] compare the two concepts vulnerability assessment and penetration testing. A brief overview of the two concepts is given and the differences are highlighted. Abu-Dabaseh et al. [ADA18] provide an overview of the two concepts of automated penetration testing and manual penetration testing and compare them in various aspects, such as efficiency, costs and completeness of the vulnerabilities that were found. Shah [Sha20] gives an overview of the two concepts automated penetration testing and manual penetration testing and examines automated penetration testing in more detail. The tools OWASP ZAP, Burp Suite, Nikto2 and Arachni are presented and compared. An overview of the two concepts of vulnerability management and automated penetration testing in combination is given by Shah and Mehtre [SM15]. They present various tools that can be used for the execution of vulnerability scans. Differences between internal and external scans are highlighted. The article details the VAPT approach, in which vulnerability management and manual penetration testing are combined. Khera et al. [KKSG19] also focus on the VAPT approach to highlight the importance of cybersecurity measures in today's society. They draw similar conclusions to [SM15].

2.3 Why this is relevant

The "Cybercrime Bundeslagebild 2022" [BKA23] and the most current "Cybercrime Bundeslagebild 2023" [BKA24] (Cybercrime Federal Situation Report) were used to illustrate the threat situation posed by cybercrime and financial damage caused by cyberattacks. The last years have seen a steady increase in cybercrime internationally and damages resulting from cyberattacks have become existence threatening for many organizations. Data from the "ENISA Threat Landscape 2023" [LTT+23] report and the "Die Lage der IT-Sicherheit in Deutschland 2023" [BSI23] ("Situation report on cybersecurity in Germany 2023") report from the Federal Office for Information Security of Germany shows, that ransomware attacks are the prime risk for organizations nowadays. Most of the targets of cybercrime are small to medium-sized businesses, so it is important to develop an efficient and working cybersecurity strategy for every organization.

2.4 Methods and tools

Sharma et al. [SSN23] compare different scoring systems for prioritizing vulnerabilities. The paper also presents its own scoring system with a focus onto the environment and the expected impact of a vulnerability. These metrics are also compared to the current CVSS versions up to version 3. The work highlights the importance of such additional metrics, even the new CVSSv4 implements similar new metrics. Walkowski et al. [WOS21] investigated the prioritization of vulnerabilities and presents different scoring systems that can be used for the prioritization of vulnerabilities. The authors also use different test environments for scanning for vulnerabilities with the Nessus scanner and show that different scoring systems impact the predicted amount of time that has to be spent to fix vulnerabilities found on the systems. They show, that the right prioritization can accelerate the process of removal of the critical vulnerabilities that were detected.

3 Methodology

The modeled organization situations are described in this chapter. An existing internal test network, as shown in Section 3.2, for testing vulnerability scanners was extended and used to run the tests. The test network was split into three different sizes with the first size containing just three of the available hosts, the second size jumping to 7 hosts and the last size doubling that to 14 hosts. With each size new and more enterprise focused services get introduced into the network. Following that, Software and operating systems are detailed and initial system configurations like usernames and passwords are described. Thereafter, the tools used for scanning the networks are described, as well as configurations for each scanner.

3.1 Organization models

At first, as modelled by the state **new**, the network only consists of three hosts, two of which are client systems and one server. This setup should mimic a newly formed organization with two employees and one simple webserver running WordPress. It includes the essential services needed to run the business. The organization uses primarily free software and has no active directory domain controller or other means of central user management. This organization is primarily focused on creating and expanding its business. **Established** represents an established organization that created a working business. Two new hosts have been added, one Windows 8.1 client resembles an aging effect of the network and an Ubuntu client to diversify operating system types. This network contains a domain controller for central user management and all hosts have been joined to the domain, except for the Ubuntu host. Besides the domain controller, a second server was added for moving resources towards virtualization. This new server is based on Alpine Linux and uses Docker for containerizing applications. The WordPress site was moved to this server and the previous server was repurposed to be a development server with the XAMPP software package installed. In sum, this network consists of 7 hosts with 3 being server systems and 4 clients. At last, **advanced** doubles the number of hosts to 14 by adding yet another Windows-based server that acts as a dedicated NAS by providing SMB shares. The aging of networks is expanded upon with a Windows 7 machine and an Ubuntu 16 machine. Two additional Windows 10 hosts and a Debian 11 client have been added as well. This time, all hosts are joined into the domain with the help of the sssd service for Linux based systems. The development server has been equipped with the DVWA[1] (damn vulnerable web application) running to demonstrate the organizations first attempt at creating their own software. An Android 9 virtual machine is also added to this network as an experimental device to get insights on mobile devices in the context of possible attack vectors.

3.2 Test network

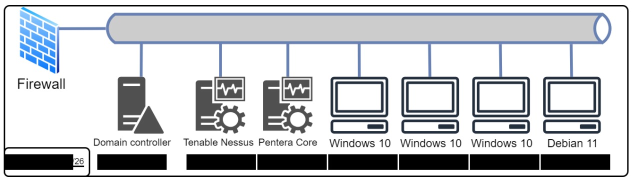


Figure 3.1: The structure of the existing test network

An existing test network for evaluating vulnerability scanners as shown in Fig. 3.1 was extended to include different operating systems and operating system versions. The original test network included Windows 10 clients only and one Debian 11 client. Extensions included a variety of Windows clients, additional Ubuntu clients and some servers as shown in Table 3.1.

Table 3.1: Available machines in the extended test network

Host	Operating system	Lifetime status
dc	Windows Server 2019	Current
win-server	Windows Server 2022	Current
ubuntu-server	Ubuntu Server 24	Current
alpine-server	Alpine Linux 3.20	Current
win-7-client	Windows 7 Pro SP1	End of support
win-8-1-client	Windows 8.1 Pro	End of support
win-10-client	Windows 10 Pro 22H2	Current
win-10-client-2	Windows 10 Pro 22H2	Current
win-10-client-3	Windows 10 Pro 22H2	Current
win-11-client	Windows 11 Pro 23H2	Current
ubuntu-16-client	Ubuntu 16.04.7	End of support
ubuntu-24-client	Ubuntu 24.04.1	Current
debian-11-client	Debian 11	Current
android-9-client	Android 9.0 R2*	End of support

**android-x86* virtual machine*

[1]https://github.com/digininja/DVWA

Together with the network states the test network represents different organization sizes. Depending on the state, a selection of systems is present in the network and additional services may be installed on the hosts. The reference date for software versions mentioned in this section is 10/30/2024.

Table 3.2: Network state "New"

Host	Services	Description
ubuntu-server	WordPress + plugins (Plugin download, WooCommerce)	Organization website + shop
win-10-client	Libre Office, RDP enabled, Steam Epic Games Launcher	Employee 1's workstation
win-11-client	Libre Office, RDP enabled, Discord, Spotify, WhatsApp, µtorrent	Employee 2's workstation

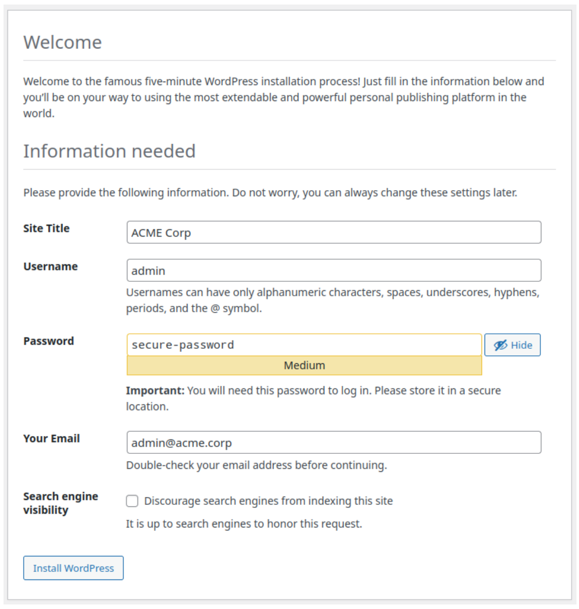
Network state "New" As the clients should not resemble purpose built work machines they have some common personal programs installed. They were both updated to the latest available version of their respective operating system.

The server represents a single webserver running WordPress with a plugin suite for running web shops. The WordPress website was set up on an Ubuntu server following a tutorial from ubuntu.com[2]. This yields a classical LAMP stack with a Linux based operating system, Apache HTTP Server as the webserver, MySQL for the database and a PHP codebase. As the new organization does not care about security at first and instead focuses on setting up a working business the server was set up with the username "user" and the password "password". The mysql password is set to "password". The tutorial states a username and password should be chosen that will only impact WordPress. So "admin" as the username and "secure-password" as the password were entered into the installation web page (Fig. 3.2).

The first result when searching the internet for a WordPress shop plugin was WooCommerce, hence it was also added to the system. A vulnerable version of the WordPress plugin download plugin was also added to the installation for convenient plugin management. No additional plugins for enhancing security were added to the WordPress installation.

The clients depict personal machines that are used for work as a newly founded organization may not have the resources yet to buy extra work computers for their employees. As such, the clients contain common consumer software.

[2] https://ubuntu.com/tutorials/install-and-configure-wordpress



Welcome

Welcome to the famous five-minute WordPress installation process! Just fill in the information below and you'll be on your way to using the most extendable and powerful personal publishing platform in the world.

Information needed

Please provide the following information. Do not worry, you can always change these settings later.

Site Title: ACME Corp

Username: admin

Usernames can have only alphanumeric characters, spaces, underscores, hyphens, periods, and the @ symbol.

Password: secure-password

Medium

Important: You will need this password to log in. Please store it in a secure location.

Your Email: admin@acme.corp

Double-check your email address before continuing.

Search engine visibility: ☐ Discourage search engines from indexing this site

It is up to search engines to honor this request.

Figure 3.2: WordPress 5 minute installation screen

All clients contain the newest available versions for their operating system and additionally installed software. The server has also gotten newest available operating system and WordPress version, including the installed plugins, except for the vulnerable plugin.

Table 3.3: Network state "Established"

Host	Services	Description
dc	Microsoft Active Directory	Domaincontroller, SMB share
ubuntu-server	XAMPP software package	Development server
alpine-server	Docker, Portainer, docker-mailserver	Web server, mail server
win-8-1-client	No further changes	Employee 1's workstation
win-10-client	Libre Office, RDP enabled, Steam Epic Games Launcher	Employee 2's workstation
win-11-client	Libre Office, RDP enabled, Discord, Spotify, WhatsApp, µtorrent	Employee 3's workstation
ubuntu-24-client	No further changes	Employee 4's workstation

Network state "Established" In this state the network has grown to include two more clients and a domain controller as well as another dedicated server for virtualization using Docker containers. Every Windows host was added to the domain controller in the domain ACME.CORP. The domain controller exposes a network share at **\\dc\data** with the default configuration for the **SMB share – fast** profile.

An Alpine Linux server was set up following the default installation procedure with no additional users added and "password" as the root password. The Docker host was set up following a tutorial from thelinuxcode.com[3]. To ease the management of Docker containers for the organization a Portainer container was added and subsequent compose stacks were managed through this Portainer instance. The WordPress containers were set up following a tutorial from digitalocean.com[4]. The only part deviating from the tutorial is the certificate configuration with certbot, a self-signed certificate is used instead. The tutorial, last updated on January 31, 2024, does not install the most recent WordPress version, but surprisingly uses version 5.1.1 released in May 2019. Besides the WordPress webserver a mailserver[5] was added to the alpine server. This project has its own documentation and was followed up to, but not including the point *Further Miscellaneous Steps*. The authoritative DNS server in this case is the domain controller and the DNS entries mentioned in the tutorial were added to its acme.corp zone. www.acme.corp also links to the alpine server. As the organization has seen that penetration testing was able to identify credentials for the ubuntu server, the password of the "user" user was changed from "password" to "secure-password". This change was made, because that is the WordPress admin user password, and it was not identified by any scanner, so it has been proven to be "secure". The organization is unaware of the dangers of reusing passwords. Because the WordPress instance was migrated to the alpine server, this Ubuntu server can now be used for other purposes. This machine will now be used as a development server and the XAMPP software package is installed.
A Windows 8.1 client was added to the network to simulate an aging network and an Ubuntu 24 client has been added to diversify operating system types.
All additional systems configured for this network have predictable usernames and weak passwords.

Network state "Advanced" A second Windows server was installed and set up to function as a dedicated NAS in the same way as the primary domain controller's SMB share. Another two Windows 10 clients were added to the network and joined to the domain. A further Windows 7 client was added to expand the aging effect of the network. For the same reason an Ubuntu 16 client was also added and joined to the domain with sssd.
The last additional Linux based system is a Debian 11 client to contribute to operating system diversity.
An Android client was also added to the network to represent the significance of mobile devices in personal and corporate environments. The Android-x86 project was used to

[3] https://thelinuxcode.com/install-docker-alpine-linux
[4] https://www.digitalocean.com/community/tutorials/how-to-install-wordpress-with-docker-compose
[5] https://github.com/docker-mailserver/docker-mailserver

realize an Android virtual machine.
All additional systems configured for this network have predictable usernames and weak passwords.

Table 3.4: Network state "Advanced"

Host	Services	Description
dc	Microsoft Active Directory	Domaincontroller
win-server	SMB shares	Dedicated NAS
alpine-server	Docker, Portainer, WordPress, docker-mailserver	Webserver and mailserver
ubuntu-server	DVWA	Testing server
win-7-client	No further changes	Employee 1's workstation
win-8-1-client	No further changes	Employee 2's workstation
win-10-client	Libre Office, RDP enabled, Steam Epic Games Launcher	Employee 3's workstation
win-10-client-2	MS Office	Employee 4's workstation
win-10-client-3	MS Office	Employee 5's workstation
win-11-client	Libre Office, RDP enabled, Discord, Spotify, WhatsApp, µtorrent	Employee 6's workstation
ubuntu-16-client	sssd*	Employee 7's workstation
ubuntu-24-client	sssd*	Employee 8's workstation
debian-11-client	sssd*	Employee 9's workstation
android-9-client	No further changes	Employee 10's work phone

**sssd is a service that can be used to integrate linux based clients into an active directory*

3.3 Tools

The comparison of the two concepts vulnerability management and automated penetration testing is carried out using two market-leading tools in these categories. Tenable Nessus Professional was used for vulnerability management and Pentera Core was used for automated penetration testing.
Tenable Nessus Professional is a network vulnerability scanner that is continuously developed to this day. It uses the nmap (Network Mapper) scanner to scan targets for open ports. The Nessus Professional product can be used as a standalone scanner or integrated into Tenable's Security Center platform. For this work the standalone variant was chosen. Tenable is the largest (by market share) company of three major players in the field of vulnerability management as can be seen in Fig. 3.3.
Tenable Nessus Professional provides a wide variety of scan types and options to configure scans. Host discovery, vulnerabilities and compliance are categories for scan types.

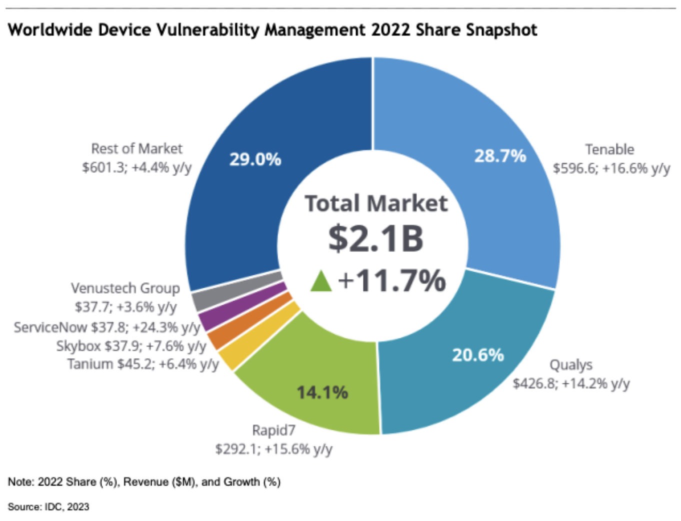


Figure 3.3: Market share of providers of vulnerability management tools

For this work the vulnerability scan type "Basic Network Scan" was chosen as it offers the least configuration options and the most preconfiguration by Tenable. Scans are split into a discovery phase and an assessment phase where discovered services are further probed. The tool has inbuilt reporting with multiple templates that focus on different aspects like vulnerabilities found, compliance or remediation of vulnerabilities.

Pentera Core is a fully automated penetration testing tool for internal network penetration tests. It takes IP ranges as its input and can dynamically expand the scope, for example to include identified Active Directory servers. Dynamic expansion was not allowed in this work. Pentera uses vulnerability scanners like OpenVAS[6] as a base to discover vulnerabilities and focus on possibly exploitable ones. The selection of vulnerabilities is then actively attacked and reported as existent if the attack was successful. If a successful attack opens the door for further vulnerabilities this cycle is repeated to gather as much information as possible. For example an SSH vulnerability may allow the tool to perform remote code execution and dig further into the systems files and services. Pentera also searches for files that may contain password hashes in standard locations or user created files with obvious names that may yield additional credentials. Actions taken by Pentera Core are categorized as achievements and may result in an exploitable vulnerability. If an action can pose significant risks for the target they have to be manually approved. Besides the above, which applies to the Black-Box testing scenario the tool offers Gray-Box testing where specific goals and additional starting information can be provided and targeted testing where scenarios like ransomware emulation can be played through on the target

[6]https://hub.docker.com/r/penpublicreps/openvas

network. When exporting the results as a report two templates can be chosen from. An executive summary and a detailed report.
Other products similar to the Pentera suite exist as well and mainly run under the term “Breach and Attack Simulation” (BAS).
To better compare the concepts and not just two products additional open source vulnerability scanners and penetration testing tools were used to examine the test network. Research for suitable FOSS alternatives to Pentera and Tenable solutions was conducted using the Google search engine as well as targeted searches on GitHub. The goal was to find free and open source network vulnerability management scanners and automated penetration testing tools.
Promising results for vulnerability management included:

- Tenable Nessus Essentials
- OpenVAS / Greenbone Community Edition

Tenable Nessus Essentials (formerly Nessus Home) is the free version of Tenable Nessus Professional. As such, it was considered too close to Tenable Nessus Professional and therefore unsuitable for a comparison.
Greenbone Community Edition is a collection of tools around the OpenVAS (Open Vulnerability Assessment Scanner). OpenVAS originated as a fork of Nessus Home, but since the fork in 2005 the two have diverged enough in development to consider them separate products and an alternative to each other. The Federal Office for Information Security (BSI) recommends using Greenbone Community Edition[7]. The specific versions of those tools are listed in Table 3.5.

Table 3.5: Used software for vulnerability management

FOSS	Name	Version	Release date
✗	Tenable Nessus Professional	10.8.3 (#10)	09/11/2024
✓	Greenbone Community Edition	23.11.0 (OpenVAS 23.8.5)	08/19/2024

Results for automated penetration testing included:

- Web application penetration testing tools like OWASP ZAP, w3af, . . .
- metasploit community edition and metasploit pro (not FOSS)
- Armitage
- Legion/Sparta
- jok3r

[7] https://www.bsi.bund.de/EN/Themen/Unternehmen-und-Organisationen/Informationen-und-Empfehlungen/Freie-Software/Tools/OpenVAS/OpenVAS_node.html

The metasploit framework, is a suite of penetration testing tools that contains a collection of reconnaissance and exploit modules. Only the paid version, metasploit pro, supports in-depth automations. However, the capabilities of the suite made it a good candidate for subsequent programs that build on metasploit community edition. It is noteworthy to point out, that metasploit offers extendability through simple batch command files called resource files and embedded Ruby code within resource files that offers much more flexibility at automating attacks. Furthermore, metasploit offers an RPC API with client libraries in many different programming languages [VO20].

Searching for tools that build upon metasploit yielded the following results: AutoSploit[8], AutoXploit[9], metasploit-autopwn[10], Mosquito[11], auto-msf[12], MapSploit[13], Quick_sploit[14], MSF-EXPLOIT[15], autoMetasploit[16], Shennina[17], deep_exploit[18], mushikago-femto[19], penta[20], Armitage[21].

Most of these tools make no serious attempt at leveraging the full capabilities of the framework. Multiple were essentially wrappers for the Shodan.io API or comparable services that try to find devices connected to the internet and just bombard them with the most common exploits. A reconnaissance or probing phase is not part of them. Therefore, these tools were not applicable for this work, as this research focuses on tools for penetrating a network, so a CIDR notation target or multiple IP addresses as targets should be accepted as inputs. In many metasploit exploit modules it is actually possible to use the RHOSTS variable that targets multiple hosts. But these tools instead utilize the RHOST variable targeting a single host. These variables stand for remote hosts and remote host respectively.

Armitage is another tool building upon metasploit. It provides an easy GUI for quickly performing diverse auxiliary and exploit modules with metasploit and can be considered one of the most advanced metasploit tools. The projects seems abandoned however, as the last update to the repository was made in 2016 and the website is down since 2021. The project used and linked to in this work is a fork of a fork of the original project and also

[8] https://github.com/NullArray/AutoSploit
[9] https://github.com/Yashvendra/AutoXploit
[10] https://github.com/hahwul/metasploit-autopwn
[11] https://github.com/r00t-3xp10it/resource_files
[12] https://github.com/akr3ch/auto-msf
[13] https://github.com/CorvusCodex/MapSploit
[14] https://github.com/Madhava-mng/Quick_sploit
[15] https://github.com/isuruwa/MSF-EXPLOIT
[16] https://github.com/paulosgf/autoMetasploit
[17] https://github.com/mazen160/shennina
[18] https://github.com/TheDreamPort/deep_exploit
[19] https://github.com/PowderKegTech/mushikago-femto
[20] https://github.com/takuzoo3868/penta
[21] https://github.com/r00t0v3rr1d3/armitage

the version currently provided in the kali linux package manager repository. This fork was last updated in 2022 and consist only of compatability updates with newer software versions. Also, Armitage does not automatically perform actions, the user has to decide on the next useful action and configure parameters for the exploits. As such Armitage can be considered semi-automatic only. Therefore, Armitage is a useful tool, but not the right choice for this research's questions.

AI is thrown into the mix with tools such as Shennina, deep_exploit and mushikago-femto. These tools also use the metasploit framework for performing exploits and use AI to select the exploit module. The usefulness of AI is debatable in these cases, as even the authors of the Shennina project describe the usage of AI in their code as follows: "The problem should be solved by a hash tree without using 'AI', however, the HITB Cyber Week AI Challenge required the project to find ways to solve it through AI."[22] The proposed framework in [VO20] builds upon this realization by implementing their AI approach not through deep neural networks, but instead use a decision tree. In the end those AI tools have shown to be unsuitable for this work as they also allow just a single target.

Next up, Legion is a fork of Secforce's Sparta and a preinstalled tool on kali linux. The last update to Sparta was made in 2020. Legion forked from Sparta in 2018 and was updated until 2023. Multiple hosts or a CIDR notation can be used as inputs for scans. The tool provides a two phase approach that is also separated in its GUI. A reconnaissance phase using different nmap scan configurations in six steps; the imported targets first tested for reachability and are then scanned for vulnerable services. And an exploitation / brute forcing phase, where possibly vulnerable services can be sent to hydra for cracking credentials or are automatically scanned by subsequent tools such as nikto, SMBenum, dirbuster and more. CVEs are also automatically added to the host descriptions if a vulnerable service was detected. Legion runs unstable and crashes can be observed regularly. It describes itself as a semi-automated penetration testing tool and besides brute forcing it actually does perform further actions automatically. The instability, lack of reporting and disordered result tabs that open another tab for each run of further tools, even if that tool did not yield any results, made it unsuitable for this work.

The jok3r project is an attempt at unifying existing tools and providing a whole suite of scanners to examine a single host or multiple targets. The last update to the project was made in 2019, a fork was maintained until late 2020 and added a web UI and dependency updates. Multiple targets have to be defined in a text file following the format HOST,PORT,SERVICE per line. Here it becomes apparent that creating such a file is no easy task, in particular it is not really feasible manually. Luckily importing a nmap XML scan output is possible and provides an easy way of defining a clearly scoped scan con-

[22] https://github.com/mazen160/shennina#why-are-we-solving-this-problem-with-ai

figuration. The project consists of a comprehensive architecture including exporting the results as an HTML report. Managing tools and scanning is accessible through a CLI and analysis and reporting is available through an interactive CLI. Similarly to metasploit workspaces jok3r offers the option to save a scan configuration and results as a so-called mission for later retrieval.

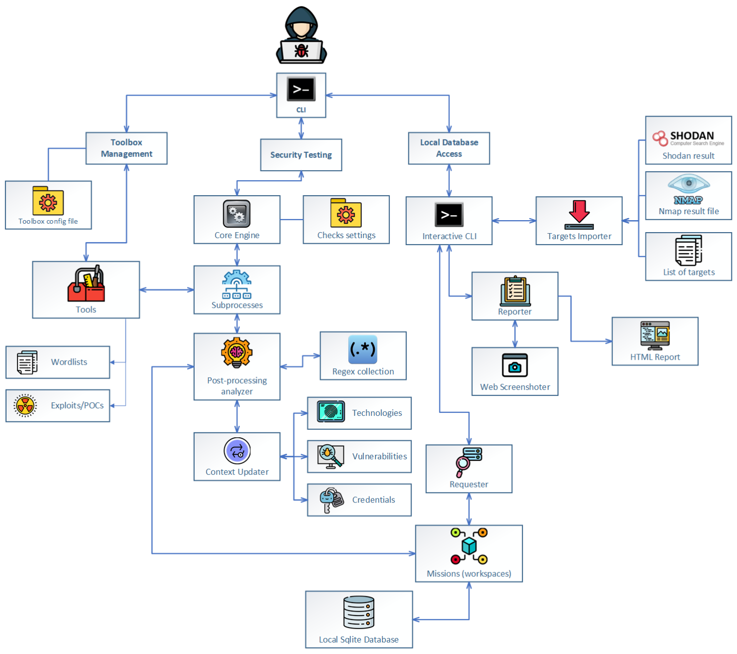


Figure 3.4: jok3r architecture (https://github.com/koutto/jok3r#id6)

Fig. 3.4 gives an overview of the structure of the jok3r tool. It is composed of three main parts; the toolbox, attacking, and database management with reporting.

Trying to run the project as is on a modern OS version fails due to outdated dependencies. However, the project is also available as a single docker image about 27 GB in size, which compared to commercial tools like Pentera Core that use a whole compose stack for the full product are surprisingly not that far apart in disk requirements. jok3r provides the necessary possibilities appropriate for this work.

Overall it proved difficult to find suitable FOSS alternatives to existing commercial automated penetration testing tools. The metasploit framework is very valuable for manual penetration testing, but extensive tools for automated penetration testing are scarce. However, jok3r also utilizes metasploit exploits as part of its toolchain. The specific versions of those tools are listed in Table 3.6.

Table 3.6: Used software for automated penetration testing

FOSS	Name	Version	Release date
✗	Pentera Core	6.2.2.2	10/2024
✓	jok3r	3 beta 2	07/10/2019

3.4 Scanning the networks

Scans were performed sequentially, so that no two tools were interfering with another. The network states were examined in the following order: new, established, advanced.

Tenable Nessus Professional The preset "Basic Network Scan" was chosen as it contains recommended settings chosen by Tenable. All default settings were used unless stated otherwise. Changes included:

```
Discovery > Scan Type: Port scan (all ports)
Assessment > Scan Type: Scan for known web vulnerabilities
Report > Designate hosts by their DNS name: Yes
Report > Display hosst that respond to ping: Yes
Report > Display unreachable hosts: Yes
```

No credentials were provided for this scan configuration.

Greenbone Community Edition The scan targets were configured with the following relevant options:

```
Port List: All IANA assigned TCP and UDP
Alive Test: Scan Config Default
Credentials: none
```

Except for the addition of UDP ports all options remain the default options chosen by the tool.
The scan task was configured with the following relevant options:

```
Min QoD: 70%
Scanner: OpenVAS Default
Scan Config: Full and fast
Order for target hosts: Sequential
Maximum concurrently executed NVTs per host: 4
Maximum concurrently scanned hosts: 20
```

All these options are the default options chosen by the tool.

Pentera Core The preset “Penetration Testing (Black Box)” was chosen and every option was left in the default state except for those below:

```
Require Approval for Exploits: no
Allow Services Bruteforce: yes
Require Approval for Services Bruteforce: no
Allow Out of IP Range Spoofing: no
Allow Automatic Active Directory Controller(s)
Identification and Queries: no
```

No credentials were added during the scan. All action approval requests during the scan were approved.

jok3r jok3r is the only tool, that requires a nmap XML output as its input. The nmap scan was performed with the following options:

```
nmap -A -sT -sU -Pn -iL input.txt -oX output.xml
```

The nmap scan targets the most common 1000 TCP and UDP ports (-sT, -sU) with vulnerability detection turned on (-A) while skipping host discovery and assuming all hosts are online (-Pn). This is a good compromise between scan speed and results. Targets are defined in an input list (-iL) and results are saved as an XML output (-oX). The findings of the nmap scan are imported into jok3r as all possible target services. No further discovery process is taking place during the attack phase. In the attack phase jok3r prompts the user for approval before executing actions such as exploits or further probing. In each scan every action proposed by the tool was confirmed and executed.
A noteworthy mention is that jok3r frequently encounters syntax errors while calling other tools. This may be related to its last update being released 5 years ago.

4 Results

Table 4.1: Amount of vulnerabilities found by each scanner

Network state	Scanner	Low	Medium	High	Critical
New	Tenable Nessus Professional	3	13	4	1
New	Greenbone Community Edition	5	6	0	–
New	Pentera Core	3	0	5	13
New	jok3r	0	9 + 1	0 + 1	–
Established	Tenable Nessus Professional	10	56	12	12
Established	Greenbone Community Edition	10	27	7	–
Established	Pentera Core	6	0	5	0
Established	jok3r*	0	20 + 5	10 + 1	–
Advanced	Tenable Nessus Professional	11	36	11	6
Advanced	Greenbone Community Edition	16	17	12	–
Advanced	Pentera Core	13	0	18	12
Advanced	jok3r**	0	26 + 8	8 + 2	–

**During the second scan jok3r reported a lot of FTP vulnerabilities, after manually checking for actually installed FTP software 1012 false positives were removed from the report. A further 5 vulnerabilities could not be categorized in severity and were added to the results in the current fractions; 3 medium and 2 high.*

***After the third scan the same 5 vulnerabilities could not be categorized in severity and were added to the results in the current fractions; 4 medium and 1 high.*

All scan results for each network state are listed in Table 4.1 and categorized by severity. Every tool uses the CVSS to categorize vulnerabilities except for jok3r. jok3r does not categorize vulnerabilities at all, but most of its underlying tools do. CMSmap is an example for this and happens to be the only tool reporting vulnerabilities for the first scan, so its evaluation from low to high is used for jok3r here instead. The second scan did also contain vulnerabilities reported by CMSmap, findings from other tools that provide a severity rating and findings from other tools that have no CVE, MS bulletin or other categorizing information assigned. The evaluation of vulnerabilities for jok3r overall is similar to the CVSSv2. Findings without a rating get added equally on top of the distribution of vulnerabilities according to the current distribution, so they don't change the ratios but count towards the amount of vulnerabilities found. Greenbone Community Edition evaluates results from low to high as well, so both tools balance each others results by using this same three-step scale. Greenbone Community Edition does use CVSSv3 categorized vulnerabilities also, but backports it into the CVSSv2 categories. This can be easily done, as CVSSv3 only introduces subdivisions of the CVSSv2 categories. jok3r does also not count gathered credentials as a vulnerability so an extra vulnerability point is awarded manually for every host and service combination where jok3r found credentials. Multiple credentials for one host and service combination count as one vulnerability. If jok3r was able to gather a username and a password, a high vulnerability was added, as medium to critical seems to be a common CVE category assignment for such vulnerabilities[23]. If jok3r was able to gather a username only, a medium vulnerability was added, as medium seems to be a common CVE category assignment for such vulnerabilities[24].

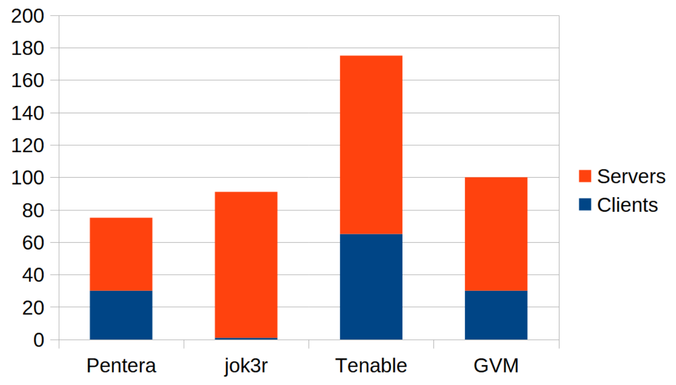


Figure 4.1: Sum of vulnerabilities found across all test networks

The vulnerability amounts found by each tool have distinctive characteristics. All tools found more server vulnerabilities than client vulnerabilities. Tenable Nessus Professional found by far the most vulnerabilities, as seen in Fig. 4.1, with 175 in total (65 client, 110 server). Pentera Core found a total of 75 vulnerabilities, 30 of which were found in clients, 45 in servers. Greenbone Community Edition detected a few more vulnerabilities at 100 in total, 30 in clients, 70 in servers. jok3r almost exclusively found vulnerabilities on servers. Just a single client vulnerability (MS17-010/EternalBlue) was detected by the tool. A further 90 vulnerabilities were detected on server systems, adding up to 91 total vulnerabilities found.

Both Pentera Core and jok3r found fewer vulnerabilities in total than their vulnerability management counterparts. Tenable Nessus Professional and Greenbone Community Edition both resemble a normal distribution, whereas Pentera Core and jok3r lean more

[23] https://cve.mitre.org/cgi-bin/cvekey.cgi?keyword=password+disclosure

[24] https://cve.mitre.org/cgi-bin/cvekey.cgi?keyword=user+enumeration

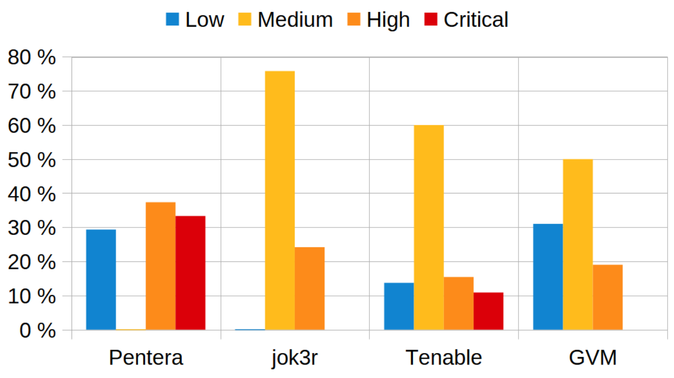
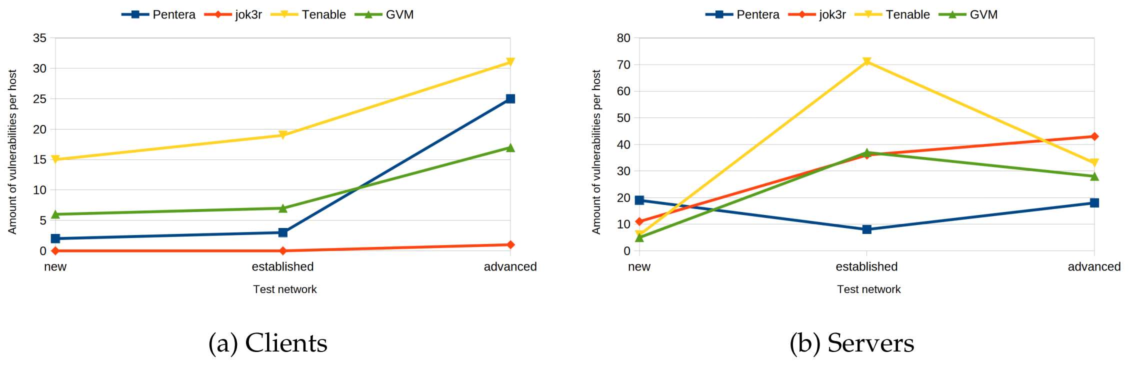


Figure 4.2: Vulnerability category distribution for each scanner across all test networks

towards higher severity vulnerabilities. This confirms the assumption that automated penetration testing focuses on high severity vulnerabilities, leaving aside vulnerabilities that likely have no or low impact and show only those that have proven to be exploitable. This is further proven in Fig. 4.2 where the distribution of vulnerabilities for each tool shows, that automated penetration testing tools indeed focuses more on high severity vulnerabilities than vulnerability management tools. Interestingly Pentera Core also shows a relatively high amount of low severity vulnerabilities against the trend set by no vulnerabilities found that are of medium severity. This can be explained by those vulnerabilities being "low-hanging fruits" that led to other higher severity vulnerabilities composing a kill chain.

4.1 Guideline



(a) Clients (b) Servers

Figure 4.3: Amount of vulnerabilities depending on network size

As the results for client and server systems are quite different as displayed in Fig. 4.3 it is not possible to create one simple formula for every scenario. The results for clients seem relatively straight forward, with a steady trend for all tools. Looking at server systems paints another picture, both concepts cross in detected vulnerabilities and change leadership throughout the experiment. A recognizable trend however in Fig. 4.4 is that vulnerabilities in hosts remain stable as vulnerability management found more vulnerabilities in clients for every network than automated penetration testing did.

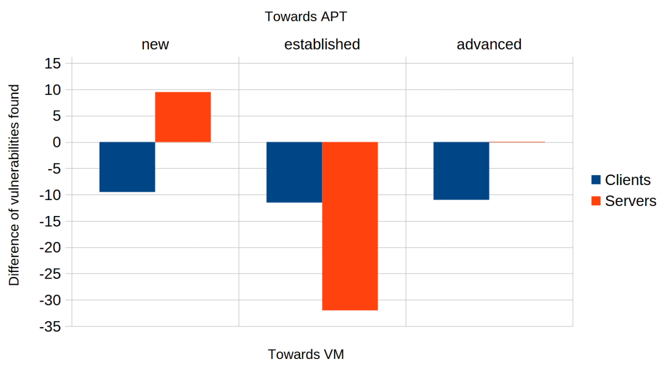


Figure 4.4: Absolute difference of vulnerabilities per concept depending on network size

When looking at Fig. 4.4 and the average vulnerability count per host in Fig. 4.5 it is easy to see vulnerabilities in servers are more volatile than clients in the **new** network and at first lean strongly towards automated penetration testing. This finding completely flips for the second network **established**. At last both approach each other to the point of being almost the same in the **advanced** network.

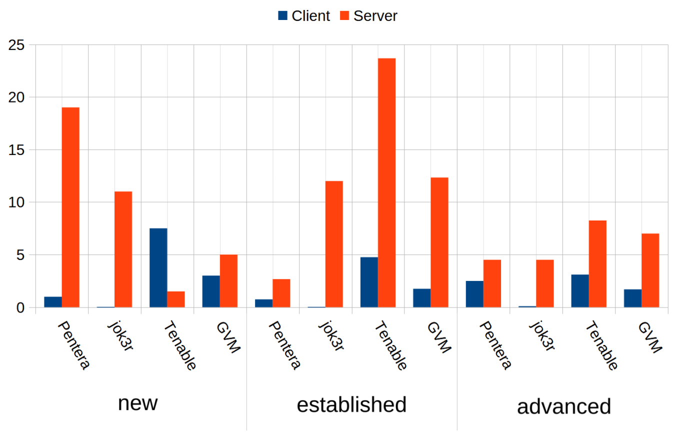


Figure 4.5: Average vulnerabilities found per host for each network

4.2 Limitations

The results from jok3r in Table 4.1 show, that the number of vulnerabilities found may still include false positives as 1012 vulnerabilities had to be removed from the results. Whenever the script for checking the success of an attack does not work reliably automated penetration can not hold up to its promise of no false positives. Automated penetration testing is a relatively new method for enhancing cybersecurity, so not many great open source tools exist for this field. But reliability will likely improve over time for those open source tools.
No great open source alternative for fully automated penetration testing tools could be found at the time of writing. jok3r was chosen, because it presents a collection of tools that work together to form a complete and automated tool for penetration testing including reporting capabilities.
The scans may not display the scanners full capabilities as all scans were left on the recommended settings for comparability. Tools such as Pentera Core build upon dynamic environments with user interaction which was not present in the test networks. DHCP attacks were not possible due to the test network being a manually configured network without a DHCP server.
As the test networks are only a single network layer, important security measures such as micro segmentation or any network segmentations at all were impossible to implement. A multi layer network can prevent attacks with adequate firewall policies.
Simulating a network may yield different results than examining an existing network from the real world.
The sample size of systems in a network may not be representative, it is difficult to manually build an extensive test network with varying systems.

4.3 Evaluation of hypotheses

When providing both approaches with the same starting point of a black-box scan, vulnerability management finds a lot more vulnerabilities than automated penetration testing. Most vulnerabilities found, for both approaches, are found in server systems. The biggest danger for clients seems to be weak credentials used by the user. The only other prominent vulnerabilities were due to outdated operating systems. Servers should therefore be considered the most critical part of the network. Users should be advised to use strong credentials and group policies should be enforced or other tools for keeping software up to date should be deployed. Interestingly no results were reported for the Android machine, so an assumption as mobile devices being a possible attack vector is not possible.
For an evaluation of what concept is better suited for an organization's network two sides

have to be considered. Client systems always display a higher detection rate of vulnerabilities using vulnerability management. So for client systems continuous monitoring using vulnerability management is the best approach. Servers on the other side are more volatile in their detected vulnerability count. When just starting to build a network, automated penetration testing finds more and obvious vulnerabilities, but after a while automated penetration testing and vulnerability management converge to the point that none finds reasonably more vulnerabilities than the other.
Automated penetration testing and vulnerability management do not have to be contrary approaches however and may be best utilized complementary. Organizations that just started building their network should use automated penetration testing for their servers and vulnerability management for their clients. Vulnerability management should be a permanent point for the organizations cybersecurity strategy. Automated penetration testing in large networks may be dropped after a while or used less frequently to save costs. If the budget is not a problem organizations should think about using both in parallel, but actual benefits are not proven and can be examined by future work.

5 Conclusions

5.1 Summary

Three different sized networks were scanned using market leading commercial vulnerability management and automated penetration testing tools and one respective alternative free and open source tool. The results show, that vulnerabilities in client systems are more often found using vulnerability management. This is a steady trend and does not change when extending the network. For server systems, the results are volatile and suggest a hybrid approach may be the best. By starting with automated penetration testing and subsequently adding vulnerability management or switching between the two after a while, most vulnerabilities will be detected. When looking at absolute numbers, it becomes apparent, that vulnerability management discovers a magnitude more vulnerabilities than automated penetration testing. This is explained through the distribution of vulnerability categories found by each approach. Vulnerability management tries to unveil every possible vulnerability in a system, even if the vulnerability just could pose a danger theoretically, but is actually prevented through other means. Automated penetration testing focuses on the most critical vulnerabilities and those that can be exploited. Prioritization of the results is handled different in each approach. Another result clearly shown in the data when looking at average vulnerabilities per host is, that servers contain the most vulnerabilities and clients significantly less. Servers should therefore be considered the most critical components of the network. Because of this, organizations should decide to use automated penetration testing at first. When the amount of client systems in the network starts growing, they should be monitored continuously using vulnerability management.

As with most of new concepts, commercial automated penetration testing tools are quite expensive and, as such, may not be an applicable solution for organizations with tight budgets. Recent reports have shown, that the risk associated with neglecting cybersecurity, especially in the beginnings of an organization, has proven to be critical in the recent past and is predicted to rise even further in the future. So investing in cybersecurity measures, either through MSSPs or trained employees, is crucial for an organization's sustained existence.

5.2 Discussion and open problems

Comparing absolute numbers may not be applicable when not using exhaustive scan configurations, as automated penetration testing only focuses on exploitable vulnerabilities and does not report existing, but non-exploitable vulnerabilities by default. Other scan configurations may yield more or less detected vulnerabilities. An approach building on this could be to count unique vulnerabilities found by one scanner but not by the other to determine which concept can extend the known attack surface more and thus reduces the entropy of possible network vulnerabilities. This would also resolve the question of using these tools in parallel. The results suggest, that switching approaches for servers at some point makes sense, but identifying unique vulnerabilities for each scan could definitively answer whether the tools overlap or complement each other in detecting the vulnerabilities.

Quality and prioritization of vulnerabilities also play a role. Finding a huge number of vulnerabilities is good in one way, but bad in terms of processing the results and designing a remediation plan, because vulnerabilities have to be prioritized. With an increasing number of vulnerabilities prioritizing them becomes more difficult. Getting a less amount, but immediate and confirmed high severity vulnerabilities could prove beneficial in this context.

An interesting question is whether real company data will continue the trend seen in this work and whether the results can thus be transferred to the real world. In this context cybersecurity maturity can be considered and taken into account when defining a trend. Also using cybersecurity maturity models could provide a better categorization and maybe even make it possible to establish a formula that gives an immediate result, instead of a general trend dependent on network growth. One drawback of this is the scope of the calculation of an organization's cybersecurity maturity score.

At the time of writing, it is safe to say that the gap between FOSS and commercial automated penetration testing tools is growing significantly as many projects seem to have been abandoned around 2020 with the project maintainers shifting their focus in other directions. An interesting project that could propel the scanning process itself is the zmap project which aims to be a faster alternative to nmap. As most, if not all, scanners use nmap for performing initial scans, replacing it with zmap could also speed up overall development and testing of vulnerability scanners. Maybe the time savings could even revive abandoned FOSS projects.

Artificial intelligence is a much discussed topic at the time and also a trend that more and more manufacturers of cybersecurity software jump into. Where exactly AI can help in the process of vulnerability management and automated penetration testing is question-able however, as the only real use case in scanners is deciding on the next best exploit to try [VO20] However, vulnerability scanners can achieve the same results by using exhaustive methods. At least the cost of running a deep neural network could turn out to be comparable to an exhaustive approach, leaving only more limited machine learning approaches such as decision tree learning. AI could indeed improve cybersecurity defenses, but may be more useful when integrated in other products such as firewalls for detecting suspicious network traffic live.

Future work can use a more sophisticated test network with multiple layers, firewalls and intrusion detection. Big corporate networks easily contain multiple ten thousand devices and many subnets and different networks overall. Many companies use a DMZ for external facing services or even an air-gapped system for critical tasks. Almost always intrusion detection is accompanied by intrusion prevention with dynamic firewall policies that interrupt a possible compromise even though the target system may be susceptible to the attack. By including these defense mechanisms in the test network it is possible to get closer to a realistic network state. However building a test network with even a thousand devices is still a tremendous task and may only be feasible by using an infrastructure as code approach.

A big oversight not only in this work, but in almost all other work is the fact that a realistic attack on a network does not encounter a static network without much traffic and changes. Some exploits are only made possible due to users interacting with services in the network making it a dynamic environment. For example phishing attacks, DNS poisoning or other disruptive tactics such as DDoS attacks on webservers or DHCP servers are not at all regarded in a low traffic environment, even if the scanners poses the capability to utilize these attack vectors. Future work could simulate realistic user behavior and corresponding traffic in the network while scanning. But fully and realistically simulating dynamic user behavior, especially interactions with webservers and attacks such as phishing attacks presents a significant challenge. In the end it comes down to the individual persons' behavior regarding whether the user falls active measures taken by the scanners.

In a large enough environment it may be possible to predict user behavior based on principles of crowd psychology. Therefore, it is important to also test the results of this work against a big enough sample of real organization data.

References

[ADA18] Farah Abu-Dabaseh and Esraa Alshammari. Automated penetration testing : An overview, 2018.

[aia19] Gartner Inc. and/or its affiliates. Gartner it score for security and risk management. Technical report, Gartner Inc., 2019.

[BKA23] BKA. Cybercrime bundeslagebild 2022. Technical report, Bundeskriminalamt, 2023.

[BKA24] BKA. Cybercrime bundeslagebild 2023. Technical report, Bundeskriminalamt, 2024.

[BRW13] Harold Booth, Doug Rike, and Gregory A Witte. The national vulnerability database (nvd): Overview. 2013.

[BSI23] BSI. Die lage der it-sicherheit in deutschland 2023. Technical report, Bundesamt für Sicherheit in der Informationstechnik, 2023.

[CSF24] *The NIST Cybersecurity Framework (CSF) 2.0.* February 2024.

[CVE] Common vulnerablities and exposures. Technical report, The MITRE Corporation.

[CVS15] Common vulnerability scoring system version 3.1. Technical report, FiRST, 2015.

[Der98] Renaud Deraison. Tenable nessus professional. Technical report, Tenable Inc., 1998.

[DH23] Nghia Trong Dinh and Vinh Truong Hoang. Recent advances of captcha security analysis: a short literature review. *Procedia Computer Science*, 218:2550–2562, 2023.

[DT15] Jignesh Doshi and Bhushan Trivedi. Comparison of vulnerability assessment and penetration testing. *International Journal of Applied Information Systems*, 8(6):51–53, 2015.

[fNS19] European Union Agency for Network and Information Security. *ENISA CSIRT maturity assessment model*. Publications Office, LU, 2019.

[GHA24] Github advisory database. Technical report, Github Inc., 2024.

[Gre07] Greenbone vulnerability management. Technical report, Greenbone, 2007.

[ISO05] Iso27001: Information security, cybersecurity and privacy protection - information security management systems - requirements. Technical report, International Organization for Standardization, International Electrotechnical Commission, 2005.

[jok17] jok3r network and web pentest automation framework. Technical report, Jérémy Brun-Nouvion, 2017.

[KKSG19] Yugansh Khera, Deepansh Kumar, Sujay, and Nidhi Garg. Analysis and impact of vulnerability assessment and penetration testing, 2019.

[LTT$^+$23] Ifigeneia Lella, Eleni Tsekmezoglou, Marianthi Theocharidou, Erika Magonara, Apostolos Malatras, RossenSvetozarov Naydenov, and Cosmin Ciobanu. Enisa threat landscape 2023. Technical report, European Union Agency for Cybersecurity, 2023.

[Mie20] Anssi Mietala. When should an organisation start vulnerability management? Master's thesis, 2020.

[MK15] Eshwar Mattadi and K Vijaya Kumar. Evaluation of penetration testing and vulnerability assessments. *International Journal of Electronics Communication and Computer Engineering*, 6(5):144–148, 2015.

[Pen15] Pentera core. Technical report, Pentera Ltd., 2015.

[Sha20] Mandar Prashant Shah. *Comparative analysis of the automated penetration testing tools*. PhD thesis, Dublin, National College of Ireland, 2020.

[SM15] Sugandh Shah and Babu M. Mehtre. An overview of vulnerability assessment and penetration testing techniques. *J. Comput. Virol. Hacking Tech.*, 11(1):27–49, 2015.

[SSN23] Abhishek Sharma, Sangeeta Sabharwal, and Sushama Nagpal. A hybrid scoring system for prioritization of software vulnerabilities. *Comput. Secur.*, 129:103256, 2023.

[Sto21] Valecia Stocchetti. Cis community defense model version 2.0. Technical report, 2021.

[VO20] Ovidiu Valea and Ciprian Oprisa. Towards pentesting automation using the metasploit framework. In Sergiu Nedevschi, Rodica Potolea, and Radu Razvan Slavescu, editors, *16th IEEE International Conference on Intelligent Computer Communication and Processing, ICCP 2020, Cluj-Napoca, Romania, September 3-5, 2020*, pages 171–178. IEEE, 2020.

[WOS21] Michał Walkowski, Jacek Oko, and Sławomir Sujecki. Vulnerability management models using a common vulnerability scoring system. *Applied Sciences*, 11, 2021.

[Yam17] Jeanne Yamfashije. Capability maturity model integration. 2017.