INSTITUT FÜR INFORMATIK
DER LUDWIG–MAXIMILIANS–UNIVERSITÄT MÜNCHEN

Bachelorarbeit

Evaluierung der Dienst- und Versionserkennung von Windows-Diensten mit Open-Source und kommerziellen Netzwerkscannern

Khrystyna Struk

Draft vom 12. Juli 2019

INSTITUT FÜR INFORMATIK
DER LUDWIG–MAXIMILIANS–UNIVERSITÄT MÜNCHEN

Bachelorarbeit

Evaluierung der Dienst- und Versionserkennung von Windows-Diensten mit Open-Source und kommerziellen Netzwerkscannern

Khrystyna Struk

Aufgabensteller:	Prof. Dr. Helmut Reiser
Betreuer:	Tobias Appel
Abgabetermin:	12. Juli 2019

Hiermit versichere ich, dass ich die vorliegende Bachelorarbeit selbständig verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel verwendet habe.

München, den 12. Juli 2019

. . . . . .
(Unterschrift des Kandidaten)

Abstract

Jede Anwendung enthält Sicherheitslücken. Um Schwachstellen auf dem Zielsystem zu identifizieren, braucht man detaillierte Informationen über die laufenden Dienste. Heutzutage gibt es zahlreiche Tools, die für die Inventarisierung des Systems und Erkennung vom Schwachstellen anwendbar sind. Einer davon ist NMAP. NMAP ist ein mächtiges Tool, das über IP-Ports den Dienst, die Versionsnummer und das Protokoll zu ermitteln versucht.
In dieser Arbeit wird aufgezeigt, wie gut NMAP und drei bekannte Schwachstellenscanner, und zwar Nessus Tenable, Rapid7 InsightVM sowie OpenVAS, bei der Diensterkennung und Versionserkennung sind.
Dazu wird ein Experiment durchgeführt. Bei der Analyse wird die Genauigkeit des Tools bei der Bestimmung des Betriebssystems, der Dienste und Versionen auf dem Zielsystem betrachtet. Als Zielsystem wird Windows Server 2016 Essential mit selbst installierten Diensten angewendet. Windows 10 Enterprise, Windows 7 Professional, Windows 8.1. werden als weitere Ziele für das Scannen eingesetzt. Um die Abhängigkeit der Dienstbestimmung von Portnummern zu ermitteln, werden die Dienste auf anderen Portnummern eingesetzt. Es wird die TCP/IP Implementierung des Windows Server 2016 geändert, um die Ergebnisstabilität der Betriebssystemidentifikation der Schwachstellenscanner zu überprüfen.
Die Daten aus dem Experiment geben eine quantitative Vorstellung über den Diensterkennungsprozess der Scanner und werden als die Basis für die Evaluierung benutzt.

Inhaltsverzeichnis

1 Einleitung

1.1 Motivation

Ein Server ist ständig einer Gefahr ausgesetzt. Angreifer und verschiedene Schadcodes stellen für den Server kritische Bedrohungen dar. Die erfolgreichen Attacken bedeuten Datenverlust bzw. unerwünschte Datenänderungen und vertrauliche Informationen können natürlich in die Hände Dritter Personen gelangen. Falls der Server mit Viren infiziert wird, kann er selbst zur Brutstätte der Infektion werden. Außerdem machen die Viren den Server langsamer und blockieren den Internet-Kanal. Auf den ersten Blick scheinen sich diese Bedrohungen in ihrer Arbeitsweise sehr voneinander zu unterscheiden. In Wirklichkeit ist dies jedoch nicht ganz der Fall. Sowohl Angreifer, als auch Schadcodes benutzen Sicherheitslücken der Software des Servers oder beliebiger Endgeräte als Angriffspunkt auf Teilnehmer im Netzwerk. Durch die Verwendung von Schwachstellen erhalten sie Zugriff auf einen Remote-Computer selbst dann, wenn der Computer gut geschützt ist. Heutzutage gibt es mehrere Quellen im Internet, wo die Schwachstellen gelistet werden. Diese Listen werden immer mit neuen Schwachstellen ergänzt. Um das Ausnutzen der Schwachstellen zu stören, muss man ständig einen Überblick über die laufenden Dienste bzw. die Software auf dem Server haben. Die Softwareversion spielt auch eine wichtige Rolle bei der Schwachstellenidentifikation. Denn nicht alle Versionen haben die gleichen Schwachstellen. Daraus folgt: je mehr Informationen über die auf den Endgeräten laufenden Dienste und deren Versionen man hat, desto leichter können die Sicherheitslücken beseitigt werden.

Ein Tool, das beim Auffinden der Schwachstellen hilft, ist ein Netzwerk Schwachstellen Scanner. Der Scanner bietet eine Reihe von Funktionen: Datenanalyse des gesamtes Systems,Identifikation der Sicherheitslücken, Bewertung, Priorisierung und Behebung der Softwareschwachstellen.

Heutzutage gibt es eine große Anzahl von Netzwerkscannern auf dem Markt, sowohl Open-Source, als auch kommerzielle. Häufig verwendete und bekannte Scanner sind z.B. NMAP, Tenable Nessus, OpenVAS Scanner, Rapid7 Nexpose, Retina usw. Sie haben unterschiedliche Arbeitsweisen und verfügen über verschiedene Methoden, um das System erfolgreich zu durchsuchen und Dienste zu erkennen.

Bei NMAP handelt es sich um ein verbreitetes Programm für Portscans, welches alle gängigen ScanTechniken beherrscht. Dabei beinhaltet NMAP nützliche Funktionen, um Dienste und Betriebssysteme sowie deren Versionsstände aufzudecken [Lex12].

1.2 Zielsetzung

Wie schon oben erwähnt wurde, ist die Diensterkennung eine wichtige Funktionalität von Netzwerk Schwachstellen Scannern. Ein Ziel dieser Bachelorarbeit ist der Vergleich von NMAP mit anderen Open-Source und kommerziellen Netzwerk-Schwachstellen-Scannern nach Funktionalität der Diensterkennung bzw. Dienstversionserkennung. Ein Ergebnis der

Arbeit ist die Analyse der Effektivität den bewerteten Tools und die Zusammenführung von Vorteilen und Nachteilen der Schwachstellen Scanner bezüglich NMAP und zu einander.

1.3 Vorgehensweise

In Rahmen dieser Bachelorarbeit sollen auf dem lokalen Windows Server bestimmte Dienste konfiguriert werden, sodass der Server als geeignete Evaluierungsumgebung für die Auswertung der Diensterkennung bzw. Versionserkennung von NMAP und der anderen Netzwerkscanner benutzt werden kann.
In diesem Experiment werden die wichtigsten und populärsten Dienste von Windows Server verwendet. Als Mitbewerber von NMAP werden Nessus Tenable , OpenVAS Scanner, Rapid7 InsightVM benutzt.
Die erhaltenen Daten aus dem beschriebenen Experiment werden die Grundlage für die Gesamtbewertung der Scanner sein. Dazu werden die Methoden untersucht bzw. beschrieben, die für Schwachstellenscanner zur Verfügung stehen, um die Diensterkennung bzw. Versionserkennung auszuführen.

1.4 Aufbau der Arbeit

Das erste Kapitel stellt eine Einleitung in das Thema dar. Es werden Motivation, Aufgabestellung und Zielsetzung der Arbeit erläutert. Kapitel Zwei beschäftigt sich mit den theoretischen Grundlagen. Es enthält eine Übersicht über wissenschaftliche Publikationen, die mit dem Thema der Bachelorarbeit verwandt sind.
Der Überblick über die Untersuchungsmaterialien, die für das Experiment benötigt werden, ist im Kapitel Drei enthalten. Zu den Untersuchungsmaterialien gehören: ausgesuchte Dienste, die Netzwerkscanner und ihre Eigenschaften. Im Kapitel Vier wird die Durchführung des Experiments beschrieben.
Im vorletzten Kapitel wird eine Interpretation der Ergebnisse bereitgestellt. Das letzte Kapitel gibt die Zusammenfassung und einen Ausblick.

2 Themenverwandte Arbeiten

Heutzutage werden viele Forschungen in Richtung der Sicherheit von Netzwerken durchgeführt. Eine umfassende Literaturrecherche liefert das Ergebnis, dass es nur eine geringe Anzahl von wissenschaftlichen Arbeiten gibt, deren Forschungsschwerpunkt das Thema des Vergleiches und der Bewertung von Schwachstellenscannern ist. In diesem Kapitel werden Publikationen vorgestellt, die für das Thema des Vergleichs der Netzwerk Schwachsteller Scanner verwandt und bedeutsam sind. Alle ausgesuchten Arbeiten sind in zwei Gruppen nach Vergleichskriterien der Scanner, die in den Publikationen überprüft wurden, angeordnet.

2.1 Vergleich nach Fähigkeit der Schwachstellenerkennung, Suchfunktion und Scan Zeit

In der Veröffentlichung „A Performance Comparison of Vulnerability Detection between Netclarity Auditor and Open Source Nessus“[SC12] beschreiben die Autoren den Leistungsvergleich nach den drei genannten Kriterien der Scannerhardware NetClarity mit der Software Nessus. Im Laufe der Arbeit wurden zwei Versuche mit gleichen Anforderungen auf zwei verschiedenen Systemen unternommen: auf dem Netzwerk der Rangsit Universität und auf dem Netzwerk der Royal Thai Army. Bei der Auswertung der Fähigkeit der Schwachstellenerkennung wurden leider keine Informationen über die Quellen von Vulnerabilität gegeben. Hier wurden die erkannten Schwachstellen mit ihren Schweregrad nach Klassifikation von NetClaritys[SC12] benutzt. Das Ergebnis der Arbeit lautet: NetClarity Adminstrator ist stärker bei der Schwachstelleerkennung und bei der Suchfunktion, aber die Scan Zeit von Nessus ist kürzer als von NetClarity Auditor.

Die Publikation von phD.Kishe R. „Comparative study of vulnerability scanning tools: Nessus vs Retina “ [R.17], die 2017 veröffentlicht wurde, handelt von dem Vergleich der zwei Schwachstellenscanner Nessus und Retina. Die beiden Tools sind teilsweise Open-Source und gelten als die am häufigsten verwendeten und bei Administratoren beliebtesten Schwachstellescanner. Die Analyse wurde mit den gleichen Prinzipien wie in der Arbeit [SC12] aufgebaut. Während des Experiments wurden auf konfigurierten Netzwerk (bzw. Testumgebung) die Leistungen der Netzwerkscanner erforscht. Es wurde nur die gesamte Aussage über die Funktionalität der Schwachstellenerkennung von beiden Tools in der Publikation erwähnt. Alle im Experiment erkannten bzw. nicht erkannten Schwachstellen von Tools wurden nicht genannt, sondern nur nach die Schweregrad der Gefährdung und Bewertung der Exposition sortiert. Nach dem Experiment wurden angemerkt, dass die beiden Tools sich bei der Identifizierung von Schwachstellen gut gezeigt haben. Die Geschwindigkeit beim Suchen (ohne Webanwendungsfunktionalität) von Nessus war wesentlich schneller als bei Retina. In Bezug auf die Scantiefe hat Nessus einen kleinen Vorteil: Nessus enthält ein Web-Mirroring-Tool, das sehr hilfreich im HTTP ist.

2.2 Vergleich nur nach Fähigkeit der Erkennung der ausgesuchten Schwachstellen

Die Masterarbeit von Johan Nilsson „Vulnerability scanner“[Nil06] handelt von empirischen Untersuchungen der Scanner und der Bewertung ihrer Tätigkeit bei der Schwachstelleerkennung. In der Publikation wurde das durchgeführte Laborexperiment und seine Ergebnisse veröffentlicht. Im Experiment wurden die bei Netzwerkadministratoren beliebtesten Schwachstellescanner (Nessus, Retina, Netrecon, ISS) benutzt, um ihre Effektivität bei weit verbreiteten Schwachstellen (RPCBIND, Finger SSH, DCOM, SSH, WWW(cmd.exe), XSS, LSASS.EXE, SQL SQL-Vorauthentifizierung, IIS.printer) zu erkennen. Offensichtlich war die Rate der Schwachstellenerkennung sehr niedrig. Kein Schwachstellescanner hat alle gegebene Schwachstellen erkannt. Retina hatte die höchste Anzahl erkannter Sicherheitslücken und hat jeder gefundenen die richtige Priorität zugeordnet, aber übersehende Schwachstellen konnten weiter durch Angreifer ausgenutzt werden. Nessus benutzt eigene Anwendungsscanner erfolgreich, und das bringt Nessus einen Vorteil. Der Autor meinte, dass die Hersteller diese anderen Scanneranwendungen als Upgrade oder als eigenständiges Produkt verkaufen. Er behauptet, dass die Schwachstellescanner nicht effektiv genug sind und man sie nicht unbedingt benutzen muss.

Der Grund, dass die Rate der Schwachstellerkennung niedrig war, könnte natürlich sein, dass die Scanner nicht auf dem neuesten Stand aktualisiert wurden. Man muss dazu sagen, dass die Arbeit 2006 veröffentlicht wurde. Höchstwahrscheinlich sind die Schwachstellscanner weiterentwickelt worden und können effizienter und effektiver als vor zehn Jahre arbeiten.

Vulnerabilities	Nmap	Nessus	Acunetix WVS	Nikto	BurpSuite
SQL Injection	√	√	√		√
Improper Error Management	√	√	√		√
Cross site Scripting	√	√	√	√	√
Rogue Servers	√	√		√	
Denial of Service	√	√	√		√
Remote Code Execution		√			
Format String Identifier		√	√		√
IIS.printer		√	√		√
DCOM					

Abbildung 2.1: [SBG14]

Im Jahr 2014 wurde eine weitere Publikation[SBG14] veröffentlicht, in der die Schwachstellerkennung von NMAP, Nessus, ACUNETIX WVS, NIKTO, BURPSUITE verglichen wurde. Die Schwachtellen, die in der Studie behandelt wurden, sind SQL-Injection, Fehlerhaftes Fehlermanagement, Cross Site Scripting, Rogue-Server, Denial-of-Service, Remo-

te Code Ausführung, Format-String-ID, IIS.printer, DCOM. Die Autoren behaupten, dass verschiedene Scanner verschiedene Arten von Schwachstellen erkennen, jedoch kein einziges Werkzeug ist in der Lage, alle Arten von Schwachstellen zu erkennen. Nessus ist der Scanner, der am meistens Schwachstellen erkannt hat, gefolgt von ACUNETIX WVS und BurpSuite. BurpSuite enthält viele Funktionen, die in anderen Tools nicht verfügbar sind und kann in andere Werkzeuge integriert werden.Die Ergebnisse des Experiment wurde in der Tabelle 2.1 angegeben:

Obwohl Netzwerkscanner zwar ein wesentlicher Bestandteil sind, um den Schutz von Netzwerken effektiv zu gewährleisten, werden sie nur stichpunktartig erforscht. In den betrachteten Forschungen wurde besondere Aufmerksamkeit auf die Fähigkeit der Schwachstellenidentifikation von Scannern gelegt. Für die erfolgreiche Erkennung ist es wichtig, die Information über die Quelle von Sicherheitslücken zu besitzen. Leider wurde die Diensterkennung bzw. Dienst Versionskennung als wichtiger Teil des Prozesses der Schwachstellenerkennung nicht betrachtet.

3 Stand der Forschung und Theorie

Obwohl OS Identification und Service Identification hier getrennt betrachtet werden, gehören sie streng genommen zusammen. Das hat damit zu tun, dass bestimmte Anwendungen exklusiv auf einem Betriebssystem eingesetzt werden. Ein Beispiel: Angenommen, ein Exchange-Server von Microsoft ist hinter einer Linux-Firewall versteckt, dann würde eine OS Identification das Linux als Betriebssystem erkennen. Eine Service Identification würde allerdings den Exchange-Server erkennen. Exchange gibt es aber nur für Windows. Und deshalb muss das Betriebssystem ein Windows sein, dass sich hinter einer Linux-Firewall befindet.Das heißt, OS Identification und Service Identification müssen zwingend kombiniert werden, damit fehlerhafte Ergebnisse reduziert werden. Und dass wir es noch mit einer Firewall zu tun haben [Wit19a].

3.1 OS-Fingerprinting

Eine bedeutsame Funktion eines guten Schwachstellenscanners ist die Bestimmung der laufenden Betriebssysteme auf dem analysierten Endgeräten.
Die Information über das Betriebssystem auf dem Gerät im Netz können mit verschiedenen Techniken bekommen werden. Die Netzwerk Schwachsteller Scanner benutzen das sogenannte OS-Fingerprinting. Unter dem Begriff des ***OS Fingerprinting*** versteht man die Erkennung von Betriebssystemen durch die Beobachtung diverser Reaktionsweisen sowie Charakteristika der sich im Netz befindlichen Systeme aus der Ferne [Wal16].
Beim OS Fingerprinting werden TCP/IP-Pakete mit bestimmten Flags an ein Ziel geschickt und die Antworten analysiert, so dass durch einen Abgleich mit einer Datenbank auf das verwendete Betriebssystem geschlossen werden kann. Die Datenbank enthält die Reaktionen von verschiedenen Betriebssystemen auf diese TCP/IP-Pakete, die versuchsweise ermittelt wurden. OS Fingerprinting nutzt den TCP/IP-Stack aus, die charakteristische TCP/ IP Implementierung, die sich von Betriebssystem zu Betriebssystem unterscheidet [AC04].

Auf der Basis, welche Pakete auf das Zielsystem geschickt werden, um das OS zu identifizieren, können alle Techniken nach Typpaket-Anfrage unterschieden werden. Alle angegebene Fingerprinting Methoden werden aus der Arbeit von Netzsicherheit und Hackerabwehr Seminar WS07/08 genommen [DD08].

TCP-Anfragen: Bei diesen Anfragen werden TCP-Pakete eingesetzt. Die Pakete enthalten verschiedene gesetzte Flags und die Reaktionen des Zielsystems auf diese Flags werden analysiert [DD08]. Die Flags befindet sich im Header des TCP-Paketes [RL81]. Auf dem Bild 3.1) wird die Struktur der Header gezeigt.

Im Weiteren werden die Methoden angegeben [DD08]:

FIN Probe
Es wird ein Paket mit gesetztem FIN-Flag an einen offenen Port gesendet. Die korrekte Antwort wäre es, nicht zu antworten, aber viele Implementierungen senden ein RST zurück.

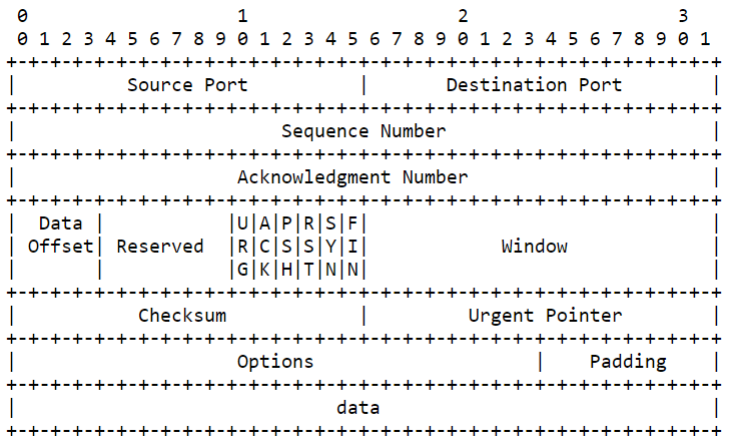

Abbildung 3.1: [RL81] TCP Header

FIN Probe ist ähnlich zum TCP FIN Scan, der jedoch davon ausgeht, dass nur geschlossene Ports ein RST zurucksenden. Die FIN Probe liefert also nur Informationen über eine Implementierung, wenn man aufgrund eines anderen Scans mit Sicherheit sagen kann, dass der angefragte Port offen ist.

TCP ISN Sampling
Hier wird versucht, die Initial Sequence Number der TCP Implementierung beim Verbindungsrequest festzustellen. Je nach Betriebssystem werden hierbei unterschiedliche Verfahren eingesetzt. Ältere Unix Systeme erhöhen in 64k Schritten, neuere erhöhen zufällig, Linux Systeme wählen zufällige Nummern und Microsoft benutzt ein zeitabhängiges Verfahren, bei dem die ISN in jeder Zeiteinheit um einen festen Betrag erhöht wird.

TCP Timestamp
Hier wird der Wert der TCP Timestamp Option, ein Feld mit der aktuellen Zeit des Senders, untersucht. Manche Implementierungen unterstutzen die Option nicht, während andere den Wert in festen Zeitintervallen erhöhen.

TCP Options
Hier werden in den versendeten Paketen bestimmte TCP Optionen benutzt. Je nachdem, ob diese auch in der Antwort enthalten sind, kann man erkennen, welche Optionen unterstützt werden und somit auf die Implementierung schließen. In einem Paket können mehrere Optionen gesetzt werden und somit gleichzeitig getestet werden.

TCP Initial Window
TCP Initial Window uberprüft die gesetzte Fenstergröße bei zurückgesendeten Paketen. Die Fenstergröße ist teilweise eindeutig einer Implementierung zuzuordnen.

ACK Value
Die ACK Flag wird jeweils als Antwort auf ein FIN/URG/PSH gesetzt und das Acknowledgement Number Feld enthält je nach Implementierung eine andere Nummer. Die meisten

Systeme setzen als Antwort auf eine FIN/PSH/URG Anfrage die Initial Sequence Number (ISN) des eingegangen Pakets als Acknowledgement Number, manche Implementierungen erhöhen die ISN um 1.

ICMP-Anfragem sind Anfragen mittels des ICMP-Protokolls. Das ICMP-Protokoll wird zum Austausch von Fehler- und Kontrollnachrichten eingesetzt [DD08].
Information in gesendete Nachricht hat folgende Struktur 3.2).

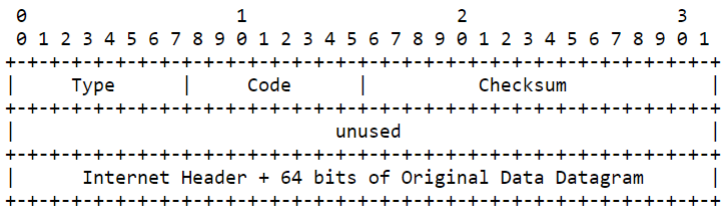

Abbildung 3.2: [Pos81a] TCP Header

ICMP-Anfragen [DD08]:

ICMP Error Message Quenching
Einige Betriebssysteme limitieren die Senderate für Fehlerbenachrichtigungen. Getestet wird dies, indem mehrere Pakete an einen zufälligen hohen UDP Port gesendet werden, der geschlossen ist. Dies provoziert ICMP Destination Unreachable Fehlerbenachrichtigungen. Anhand der Eingangsrate der Fehlerbenachrichtigungen des gescannten Systems kann die Implementierung dann identifiziert werden.

ICMP Message Quoting
Bei einer ICMP Fehlerbenachrichtigung wird ein Teil des fehlerverursachenden Pakets wieder zurückgesandt. Die Implementierungen unterscheiden sich hier in der Menge der mitgesendeten Daten. Fast alle Systeme senden den IP-Header und acht Bytes zurück, Solaris- und Linux-Betriebssysteme senden ein Bit bzw. noch mehr Daten mit zurück.

ICMP Error Message Echoing Integrity
Diese Methode funktioniert wie ICMP Message Quoting, auch hier werden die ICMP Fehlerbenachrichtigungen untersucht. Hier werden speziell das Headerfeld und die Prüfsumme analysiert.

Type of Service
Das Type of Service Feld in der ICMP Fehlerbenachrichtigung ICMP Port Unreachable ist standardmäßig auf 0 gesetzt. Manche Implementierungen weichen in diesem Punkt jedoch ab.

IP-Anfragen
Bei IP-Anfragen werden Felder des IP-Headers [Pos81a] und spezielle Eigenschaften der jeweiligen Implementierung im Bezug auf Fragmentierung untersucht[DD08]. Hier wird die IP Header aus RFC 3.3) angegeben.

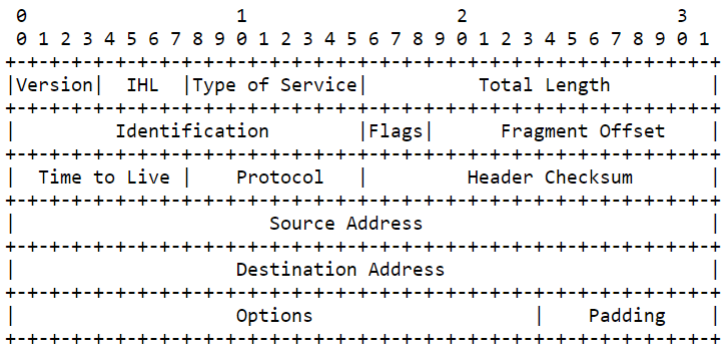

Abbildung 3.3: [Pos81b] IP Header

IP-Anfragen[DD08]:

IPID Sampling
Hier wird das IP Identification Feld analysiert. Das Identification Feld im IP-Header enthält eine Nummer, die hilft, fragmentierte Teile eines Datagramms korrekt zu reassemblieren. Die meisten Betriebssysteme erhöhen die IP ID mit jedem versendeten Paket um 1, während andere die ID zufällig wählen oder nur in festen Schritten größer als 1 erhöhen.

Don't Fragment Bit
Das Don't Fragment Bit im IP-Header verbietet das Fragmentieren des Datagramms. Einige Systeme setzen dieses Bit in bestimmten Fällen, während andere es gar nicht setzen, so dass das jeweilige Verhalten helfen kann, die Implementierung zu identifizieren.

Fragmentation Handling
Diese Technik untersucht die Reassemblierung von fragmentierten IP-Paketen. Im Speziellen wird hier die Reassemblierung von überlappenden Fragmenten beobachtet. Je nach Implementierung besitzt der ältere oder der neuere überlappende Teil Gültigkeit

Ein anderer Ansatz für die Analyse von Betriebssystemerkennung ist das sogenannte ***Banner-Grabbing***.

Mit dem Banner-Grabbing kann man eine Verbindung öffnen und das Banner oder die Antwort lesen, die von der Anwendung gesendet werden. Viele E-Mail-, FTP- und Webserver antworten auf eine Telnet-Verbindung mit dem Namen und der Version der Software. Dies hilft einem Hacker beim Fingerprinting des Betriebssystems und der Anwendungssoftware. Ein Microsoft Exchange-E-Mail-Server würde beispielsweise nur unter Windows installiert [ano11].

Jede der oben genannten Methoden bzw. Techniken ist Teil von des Prozesses der Betriebsystem Identifikation. Das Verfahren für die Bestimmung der Betriebssysteme heißt OS-Fingerprinting. Abhängig von den eingesetzten Methoden unterscheiden sich die Schwachstellen-Scanner nach ihrer Effektivität und Effizienz.

3.2 Diensterkennung

Ziel der Dienstidentifikation ist, die Anwendung und deren Version möglichst genau zu bestimmen. Dann kann man mit diesen Informationen gezielt nach bekannten Schwachstellen suchen. Eine genaue Versionsnummer hilft enorm beim Ermitteln eines geeigneten Exploits, für den das Zielsystem anfällig ist. Sofern die Anwendung diese Informationen nicht direkt herausrückt, besteht die Möglichkeit, Anwendungs-Meldungen und Protokoll-Eigenheiten auszuwerten und mit einer Signaturdatenbank abzugleichen [Wit19a].

Es ist selbstverständlich, dass die Information über laufende Dienste auf dem Zielsystem nicht nur für Angreifer nützlich ist, sondern auch für den System Administrator. Jedes System besitzt viele Dienste, die nicht aktuell sind oder überhaupt unbenutzbar. Falls der System- und Netzwerk-Administrator einen Überblick über den Server hat, kann er ein Update durchführen, Risiko über mögliche Angriffe schätzen und unnötige Dienste abschalten. Dies kann wertvolle System-Ressourcen sparen und das Sicherheitsniveau des gesamten Systems erhöhen.

Vor dem eigentlichen Prozess der Diensterkennung folgt üblicherweise ein Port Scan, um offene Ports, hinter denen sich die zu analysierenden Dienste verbergen, aufzuspüren. Hier kommen viele Methoden zum Einsatz, deren Ziel ist, die Verbindung zu den Ports aufzubauen, um sicher zu stellen, dass die Ports erreichbar sind bzw. offen. Also wird die Erreichbarkeit dabei über den Zustand der Ports bestimmt.
Grob gesehen gibt es drei Arten von Zuständen [Wit19b]:

OPEN (offen) bedeutet, dass auf diesem Port eine Anwendung oder ein Dienst lauscht und somit eine Verbindung möglich ist.

CLOSED (geschlossen) bedeutet, dass vom Host eine Verbindung auf diesen Port abgelehnt wurde. Dahinter lauscht keine Anwendung.

FILTERED/BLOCKED(gefiltert/blockiert) bedeutet, dass der Host auf Anfragen auf diesen Port nicht reagiert. Das heißt, die Verbindung wird weder bestätigt (OPEN) noch abgelehnt (CLOSED). Wenn der Host ansonsten erreichbar ist, dann findet eine Filterung (FILTERED) bzw. Blockierung (BLOCKED) durch eine Firewall statt. Das kann zum Beispiel ein vorgeschaltetes System oder ein Paketfilter auf dem Host sein.

Beim Port-Scan interessiert eigentlich nur, ob ein Port offen oder geschlossen ist. Der gefilterte Zustand ist ärgerlich, weil er weitere Untersuchungen in Form weiterer Port-Scans nach sich zieht [Wit19b].
Mit Hilfe dieser Information kann nun begonnen werden, die Prozesse hinter den offenen Ports zu analysieren. Die bekannten Methoden der Diensterkennung sind: ***Applikation-Mapping, Bannergrabbing, Service-Fingerprinting***.

Application-Mapping.

Der Ansatz des Application-Mappings, Spekulationen über das Protokoll auf TCP/IP Anwendungsschicht hinter einem offenen Port zu betreiben, ist der Vergleich mit offiziellen Port-Mappings. Von IANA werden offiziell und standardisiert Portsnummern für bestimmte Dienste vergeben. Die Liste der standardisierten Ports liegt öffentlich und zugänglich im

Internen [ano19] und kann von jedem angeschaut werden.
So ist es sehr wahrscheinlich, dass sich hinter Port 80 ein Webserver befindet, denn gängige Web-Browser verwenden standardmäßig diesen Port, um sich zu einem Webserver zu verbinden. Die Anwendung von Application Mapping hat vermutlich hohe Erfolgsquoten, kann aber leicht ausgehebelt werden. Ein Netzwerk Administrator könnte etwa bestimmte Dienste absichtlich auf unerwarteten Portnummern horchen lassen, um mögliche Angreifer zu verwirren[Gla08].

Vor allem aber lässt sich auf diese Weise nur das zugrundeliegende Anwendungs-Protokoll erkennen, nicht jedoch ein konkreter Dienst oder gar ein bestimmtes Release dieses Dienstes. Dies sollte jedoch das Ziel eines erfolgreichen Fingerprintings sein, um den passenden Exploit für den jeweiligen Dienst auswählen zu können. Schließlich ist dieser Ansatz untauglich, wenn sich hinter geöffneten Ports nicht bei der IANA registrierte Dienste und insbesondere Individualsoftware-Produkte verbergen [Gla08].

Bannergrabbing

Viele Dienste identifizieren sich beim Verbindungsaufbau mit einem Begrüßungstext. Die sogenannten Banner enthält für den Anwender eine kurze Information über Service und Host. Deshalb bietet es sich für einen Angreifer an, auf Anwendungsebene eine Verbindung vorzutäuschen, um an diese Banner zu kommen und sie auszuwerten. Beim Bannergrabbing braucht man neben der IP-Adresse des Dienstes auch dessen Portnummer auf die man sich verbindet, was man im Regelfall mit einem Port-Scan herausfinden kann. Mit etwas Glück gibt der Dienst bei der Verbindungsaufnahme ein paar Informationen über sich preis [Wit19a]. Folgendes Beispiel soll die Wirksamkeit eines gezielten Banner-Grabbings verdeutlichen 3.4):

```
~ $ telnet hackthissite.org 80
Trying 207.210.114.39...
Connected to hackthissite.org.
Escape character is '^]'.
GET /index.html HTTP/1.1
Host: www.example.net

HTTP/1.1 301 Moved Permanently
Date: Sun, 06 Jul 2008 03:05:02 GMT
Server: Apache/2.2.4 (FreeBSD) mod_ssl/2.2.4 OpenSSL/0.9.8e DAV/2 PHP/5.2.3
with Suhosin-Patch mod_perl/2.0.3 Perl/v5.8.8 [...]
```

Abbildung 3.4: [RL81] Willkomen Banner

Dieser Webserver verrät nicht nur die eingesetzte Server-Software, sondern auch Details über die spezifische Version, verwendete Module, Patches und das zugrunde liegende Betriebssystem. Dieses Beispiel macht auch deutlich, dass Banner-Grabbing ebenfalls zur Erkennung des Betriebssystems eingesetzt werden kann [Gla08].

Auf die Informationen, die man durch Bannergrabbing erhalten hat, darf man sich aber nicht verlassen. Ein sehr guter Systemadministrator weiß um die Bedeutung für den Angreifer. Deshalb sollte der Angreifer in jedem Fall die Informationen zusätzlich prüfen, die er per

Bannergrabbing erhalten hat. Es kann sein, dass der Systemadministrator den Banner manipuliert hat, um eine andere Applikation oder sogar einen anderen Service vorzutäuschen. Die ausgegebenen Informationen sind in jedem Fall in Frage zu stellen [Wit19a].

Service-Fingerprinting

Ziele des Service-Fingerprintings: Anwendungsprotokoll zu ermitteln, die verwendete Server-Software herauszufinden und genaue Release zu identifizieren.
Dafür müssen wie bei der TCP/IP-Stack-Analyse entsprechende Testfälle erstellt, über das Netzwerk versandt und die Antworten-Nachricht gesammelt und analysiert werden. Service-Fingerprinting-Software wird meist unter der Annahme entwickelt, dass das Anwendungsprotokoll bereits bekannt ist. Da jedes Protokoll der Anwendungsschicht ein anderes Vorgehen erfordert, kann es keine universelle Service-Fingerprinting-Software geben, es sei denn, als Zusammenstellung verschiedener Fingerprinting-Ansätze. Mit der Kenntnis des Anwendungsprotokolls können speziell präparierte Nachrichten an das Zielsystem und den Zielport gesendet werden. Diese Nachrichten zeichnen sich dadurch aus, dass sie das jeweilige Anwendungsprotokoll grundsätzlich einhalten, jedoch gewisse (dem Protokoll konforme oder vom Protokoll abweichende) Änderungen vornehmen, die zu einem unterschiedlichen Antwort-Verhalten der Zielanwendungen führen. Aufgrund dieser Unterschiede im Antwort-Verhalten können daraufhin Fingerprints der jeweiligen Dienste erstellt werden, die zur späteren Identifikation des gleichen odereines ähnlichen Dienstes dienen [Gla08].

Einige Fingerprints der jeweiligen Dienste sind schon im einer Signaturdatenbank erhältlich und bei der Überreinstimmung kann man sicher die Dienst und seine Version zu identifizieren.

Die einzelnen Schritte des Diensterkennungsprozesses werden im Bild 3.5): abgebildet.

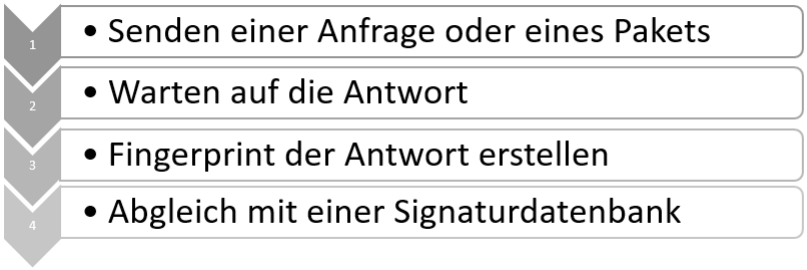

Abbildung 3.5: Ablauf des Diensterkennungprozesse

Weiter wird ein Beispiel für den Prozess des Fingerprints als Methode für die Identifikation von Webservern gegeben [Gla08]: Als verwendete HTTP-Methode kommt dabei HEAD zum Tragen, die lediglich den Header der HTTP-Antwort zurück liefert. Das hat den Grund, dass der Content der angeforderten URL nicht zur Unterscheidung dient. Schließlich können mit dem gleichen WebserverRelease verschiedene Inhalte bereitgestellt werden. Ferner wird eine HTTP-Anfrage mit überlangem URL erzeugt und über das Netzwerk versendet.

Diese Anfrage könnte z. B.folgendermaßen aussehen:

```
Request URI: http://www.edu.lmu.de/
HTTP/1.1 200 OK
Date: Sun, 07 Jul 2019 19:45:22 GMT
Server: Apache/2.2.34 (Linux)
Last-Modified: Wed, 14 Jul 2010 18:49:10 GMT
```

Die Antwort des Webservers enthält u. a. den HTTP-Statuscode, der verschiedene Situationen, die in HTTP auftreten können, charakterisiert. Für den Fall einer überlangen Antwort wären verschiedene Statuscodes denkbar: 404 Not Found, 403 Forbidden, und 414 Request-URI Too Large. Dabei werden z. B. vom Webserver Apache, je nach Release und Länge des übergebenen URLs, verschiedene Statuscodes zurückgeliefert, die zur Identifizierung des Apache-Releases dienen können. Die Informationen, die man durch das Service-Fingerprintings erhält sind in jedem Fall zuverlässiger als beim Bannergrabbing. Allerdings dauert es auch länger. Und die Qualität und Aktualität der Signaturdatenbank ist von ausschlaggebender Bedeutung. Allerdings kann ein ungewöhnliches Paketsendeverhalten die IDS zum Detektieren veranlassen oder den Service zum Absturz bringen[Wit19a].

4 Untersuchungsmaterialien

4.1 Vorgehensweise

Das Ziel der Studien wird aus dem gesamten Ziel der These abgeleitet: Evaluierung des ausgesuchten Schwachstellen Scanners nach Fähigkeit der Diensterkennung und Versionserkennung. Während des Experiments werden die Schwachstellen nach **zwei nachfolgenden Kriterien** eingeteilt:

1.Dienstidentifikation und Versionserkennung.

Hier wird überprüft, wie genau die Schwachstellen Scanner bei der Diensterkennung sind und welche weiteren Informationen über die Dienste erhalten werden können: Welches Protokoll und welche Versionsnummer hat die Anwendung, auf welchem Port läuft diese. Je mehr Informationen über die Anwendungen das Tools erfahren kann, desto genauer können die Sicherheitslücken im System identifiziert werden. Es ist sehr bedeutsam, dass die Fehlerrate bei der Dienstidentifikation nicht sehr hoch ist. Fehlerhafte Ergebnisse geben falsche Vorstellung über das System und führen zur Verwendung von zusätzlichen Ressourcen, was nutzlos und unerwünscht ist. Deswegen werden hier quantitativ nicht nur richtig anerkannte Dienste gemessen und protokolliert, sondern auch die falsch erkannten Dienste.

2. Betriebsystemerkennung.

Jeder Dienst kann nur auf bestimmten Betriebssystemen laufen. Es gibt eine Anzahl von Diensten, die nur auf bestimmten Betriebssystemen laufen können. Falls das Betriebssystem des Hosts richtig identifiziert werden kann, kann das Tool den Scanner Prozess gezielter durchführen, was natürlich eine Erhöhung der Effizienz bringt. Es wird auch folgendes Merkmal der Schwachstellenscanner untersucht: wie stabil sind die Ergebnisse bei der OS Erkennung, wie leicht kann man die Schwachstellen Scanner täuschen, sodass die Tools das richtige OS möglichst nicht erkennen können.

Es wird auch grob die benötige **Zeit für den Scanprozess** bei jedem Tool gemessen. Das ermöglicht eine Aussage über den Grad der gesamten Schnelligkeit der Schwachstellenscanner zu machen.

Im Experiment werden die unten genannten Betriebssysteme als **Zielsysteme** eingesetzt:

Windows Server 2016 Essential mit manuell installierten Diensten

Windows 10 Enterprise: Leeres System

Windows 8.1: Leeres System

Windows 7 Proffesional: Leeres System.

Die Zielsysteme werden mit den folgenden **Schwachstellen Scannern** gescannt:

NMAP 7.70

Nessus Essentials 8.4.0

OpenVAS 9

Rapid7 InsightVM

Der **Ablauf des Experiments** besteht grob aus folgenden Teilen:

1. Vorbereitung der Evaluierungsumgebung

1.1. Installation und Konfiguration der Zielsysteme
Konfiguration von Windows Server 2016 mit den ausgesuchten Diensten und die Einrichtung der virtuellen Maschinen mit weiteren drei oben genannten Betriebssystemen.

1.2.Installation und Konfiguration der Netzwerkscanner
Installation der Schwachstellenscanner und Konfiguration des Scan Modus, der für das ganze Experiment unverändert bleibt. Alle Schwachstellenscanner bekommen die gleichen Einstellungen (gleiche Anzahl Ports und Anforderungen), sodass man die Tools objektiv vergleichen kann.

2.Scans

2.1. Erster Scan: Standard
Alle konfigurierten Scanner werden für die Scans der Zielsysteme eingesetzt. Die Dienste werden sich auf den bekannten Portnummern befindet. Erwartetes Ergebnis: Erkennung der laufenden Dienste mit Versionsnummern und richtig erkannten OS Zielsystemen. Hier werden zum Test alle vier Betriebssysteme zum Einsatz kommen.

2.2. Zweiter Scan: Änderung der Portnummern
Hier werden die Portnummern der Dienste geändert. Es wird die Konfiguration unseres Servers durchgeführt: Verstecken den Dienste auf höheren Portsnummern oder Platztausch der bekannten Ports. Hier wird überprüft, ob die Portnummer der Dienste Einfluss hat, sowohl auf die Auswahl der Methoden der Schachstellenscanner im Scanprozess, als auch auf das Endergebnis. Berücksichtigt man, dass auf den Desktop Betriebssystemen nur einzelne installierte Dienste existieren, macht es Sinn, das nur auf Windows Server 2016 zu testen.

2.3. Dritter Scan: Täuschungsversuch
Mit Hilfe von „TCPOptimizer.exe“ wird der TCP/IP Fingerprint bzw. bestimmte Parameter TTL geändert, sodass die OS Erkennung für die Schwachstellenscanner schwerer wird. Es wird dazu bei den Servereinstellungen die „TCPWindows Size“ geändert.

Der Scan wird auf Windows Server 2016 durgeführt.

Die Ergebnisse der durchgeführten Scans werden als Material für die Bewertung der Schwachstellenscanner benutzt.

4.2 Schwachstellen-Scanner

Im diesem Kapitel wird eine Übersicht über die vier Netzwerkscanner gegeben, die für das Experiment ausgesucht wurden. Hier werden wichtige Merkmale der Tools präsentiert und es folgt ein Vergleich basierend auf Literaturquellen.

NMAP

Nmap(„Network Mapper“) ist ein Open-Source-Werkzeug, das für die Netzwerkanalyse und Sicherheitsüberprüfung entwickelt wurde. Die erste Version wurde schon im September 1997 von Fyodor „Lyon“ veröffentlicht und hat am Anfang nur Linux Betriebssysteme unterstützt [Rei15].

Dank einer starke Community hat das Tool, das zuerst nur aus 2000 Zeilen Code bestand, im Laufe der Jahre viel neue Funktionalität erworben und ist der populärste Network Security Scanner geworden. Basierend auf vielen Literaturquellen und verschiedenen veröffentlichen Bewertungen von Spezialisten in IT-Sicherheit (hier sind die Links), muss man schon erwähnen, dass NMAP heutzutage mehreren sowohl Open-Source als auch kommerziellen Schwachstellenscannern große Konkurrenz macht.
Die aktuelle Version von NMAP ist 7.70 ab 28.03.2018 und kann hier heruntergeladen werden [nma15].
NMAP wird heutzutage von vielen Betriebssystemen unterstützt. Das Tool ist für Linux, Windows und Mac Betriebssysteme verfügbar und kann über klassische Kommandozeile oder über eine GUI-Schnittstelle ausgeführt werden.

Zenmap ist die offizielle Grafische Benutzeroberfläche für das Tool und kann kostenlos heruntergeladen werden. Wie der Scanner kann auch die GUI-Schnittstelle auf den meisten bekannten Betriebssystemen installiert und benutzt werden. Hier kann man mit Hilfe eines Befehlserstellers interaktiv NMAP-Befehlszeilen erstellen. Scanergebnisse können gespeichert und später angezeigt werden. Das ermöglicht für dem Benutzer, die Ergebnisse mit früheren Scanergebnissen zu vergleichen. Ein sehr wichtiges Merkmal ist, dass die letzten Scanergebnisse in eine durchsuchbare Datenbank gespeichert werden. Die Datenbank erleichtert den Analyseprozess bei bestimmten Komponenten des Systems, z.B. bei kritischen oder wertvollen Diensten für die Benutzer.

Nmap benutzt rohe IP-Pakete auf neuartige Weise, um festzustellen, welche Hosts im Netzwerk verfügbar sind, welche Dienste (Anwendungsname und -version) diese Hosts bieten, welche Betriebssysteme (und Versionen davon) darauf laufen, welche Art von Paketfiltern/-Firewalls benutzt werden sowie Dutzende anderer Eigenschaften. Auch wenn Nmap üblicherweise für Sicherheitsüberprüfungen verwendet wird, wird es von vielen Systemen und Netzwerkadministratoren für Routineaufgaben benutzt, z.B. Netzwerkinventarisierung, Verwaltung von Ablaufplänen für Dienstaktualisierungen und die Überwachung von Betriebszeiten von Hosts oder Diensten [Rei15].

Dank der nmap-services Datenbank, die über 2200 bekannte Dienste verfügt, kann NMAP Aussagen über laufende Dienste auf den offenen Ports des Zielsystems machen.
Um die mehr Informationen über den Dienst zu gewinnen, werden die Scan-Methoden angewendet.

Die nmap-service-probes Datenbank enthält Testpakete für die Abfrage verschiedenster Dienste und Ausdrücke für den Vergleich und das Parsen der Antworten. Nmap versucht, nicht nur das Dienstprotokoll zu bestimmen (z.B. FTP, SSH, Telnet, HTTP), sondern auch Anwendungsnamen (z.B. ISC BIND, Apache httpd, Solaris telnetd), Versionsnummer, Hostnamen, Gerätetyp (z.B. Drucker, Router), die Betriebssystemfamilie (z.B. Windows, Linux) und manchmal verschiedene Details: etwa ob ein X-Server Verbindungen annimmt, die SSH-Protokollversion oder der KaZaA-Benutzername. Natürlich bieten die meisten Dienste nicht all diese Information. Falls Nmap mit OpenSSL-Unterstützung kompiliert wurde, verbindet es sich mit SSL-Servern, um den hinter dieser Verschlüsselungsebene lauschenden Dienst zu ermitteln [Rei15].
Unten wird es ein Beispiel von der Scann Ergebnisse angezeigt.

```
nmap -sV -T4 -F www.zhs-muenchen.de
Starting Nmap 7.70 ( https://nmap.org ) at 2019-07-05 21:41
Caiaaiay Aa?iia (eaoi)
Nmap scan report for www.zhs-muenchen.de (129.187.254.81)
Host is up (0.031s latency).
rDNS record for 129.187.254.81: wwwv2.tum.de
Not shown: 94 closed ports
PORT STATE SERVICE VERSION
21/tcp filtered ftp
23/tcp filtered telnet
25/tcp filtered smtp
80/tcp open http Apache httpd
443/tcp open ssl/http Apache httpd
```

Was natürlich sehr praktisch ist, ist dass die Benutzer die Datenbanken erweitern können. Wenn Nmap Antworten von einem Dienst erhält, aber keine Übereinstimmungen dafür in seiner Datenbank finden kann, gibt es einen speziellen Fingerprint und eine URL aus, damit Sie[Benutzer] diese Antwort einsenden können, falls Sie[Benutzer] genau wissen, was auf diesem Port läuft [Rei15].

Wie schon oben erwähnt wurde, verfügt NMAP über Betriebssystemerkennung mit TCP/IP-Stack-Fingerprinting. Nmap sendet eine Reihe von TCP- und UDP-Paketen an den entfernten Host und untersucht praktisch jedes Bit in der Antwort. Nach der Durchführung Dutzender von Tests, wie z.B. einer TCP-ISN-Abtastung, Unterstützung und Reihenfolge von TCP-Optionen, IP-ID-Abtastung und Prüfung der initialen Fenstergröße, vergleicht Nmap die Ergebnisse mit seiner Datenbank in nmap-os-db von über eintausend bekannten Betriebssystem-Fingerprints und gibt die Details zum Betriebssystem aus, wenn es eine Übereinstimmung gefunden hat [Rei15].
Bei keiner Übereinstimmung hat der Benutzer eine Option, genau wie bei der Diensterkennung, einen Eintrag in der OS Fingerprint Datenbank (nmap-os-db) erstellen. Eine weitere

zusätzliche Information, die die Betriebssystem-Erkennung aktiviert, ist eine Schätzung der Betriebszeit des Zielhosts. Dabei wird die TCP-Timestamp-Option (RFC 1323) benutzt, um zu raten, wann ein Rechner das letzte Mal neu gestartet wurde [Rei15].

Rapid7 InsagtVM

Rapid7 wurde im Jahr 2000 gegründet und hat sich im Laufe der Jahre auf Sicherheitsdaten und Analysetechnologien konzentriert, einschließlich des Schwachstellenmanagements, das Unternehmen dabei unterstützt,die Sicherheit zu verbessern. Rapid7 InsightVM ist ein Echtzeitverwaltungssystem für Sicherheitslücken und Endpunktanalysen. Getrieben durch das preisgekröntes Nexpose Produkt, nutzt InsightVM neueste Analytiken und Endpoint-Technologien, um Schwachstellen aufzudecken, priorisieren und nachzuweisen, sodass Risiken eliminiert werden[Rapb].

InsigtVM ist eine Komponente der Rapid 7 Insight-Plattform 4.1. Sie bietet eine gemeinsame Sichtweise auf Sicherheit, IT-Betrieb und DevOps. Eine Cloud-Plattform bietet mit einem Klick Zugriff auf das Schwachstellenmanagement von Rapid7[Rapb].

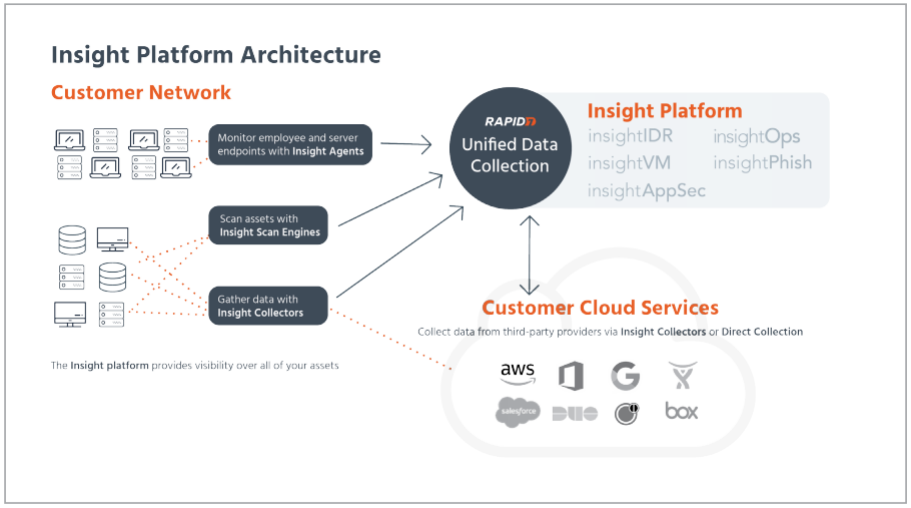

Abbildung 4.1: Insight-Platworm Architektur [Rapc]

InsightVM speichert nicht nur die Daten, sondern verbindet das Wissen aus der Schwachstellenforschung von Nexpose, die Exploit-Daten von Metasploit, Angreiferverhalten weltweit, Internet-weite Scan-Daten und Bedrohungsanalysen, um aus Daten Antworten zu liefern. Rapid7 ist der einzige Anbieter, der zur CVE-Nummernvergabe berechtigt ist [Rapb].

InsightVM enthält alle Funktionen von Nexpose Enterprise.

Wichtige Merkmale des InsightVM[Rapa]:

Leistungsstarke Analyse – Möglichkeit die erweiterten Analysen der Auswirkungen von Bedrohungen mit echten Risikoinformationen zu bekommen, um die Situation schneller zu beheben.
Kontinuierliche Überwachung - Der Analyst überwacht den Endpunkt automatisch und kontinuierlich auf Schwachstellen und bietet Live-Überwachung. Es bietet dynamische Überwachung für Azure und VMware.
Liveaboards ist ein interaktives Dashboard mit Echtzeitdaten für CISO, mit dem ein Systemadministrator die Sicherheit der Infrastruktur analysieren kann.
Risikobewertungen - Schwachstellen werden mit einer Standard-CVSS-Bewertung gekennzeichnet, sodass Sie vorrangige Maßnahmen ergreifen können.
Integration - lässt sich in Ihre bevorzugten Tools wie Metaspoilt, InsightIDR, Nexpose, ServiceNow, McAfee, Splunk usw. integrieren.

Für das Experiment wurde eine Test Version von Rapid7 InsightVM für 30 Tage benutzt.

Tenable Nessus Essentials

Nessus ist einer der am meisten verbreiteten und benutzten Schwachstellen Scanner und nach Angaben des Herstellers in mehr als 27000 Unternehmen im Einsatz.

Ursprünglich war Nessus ein Open-Source Produkt, aber seit 2005 (Version 3) wurde das Lizenzmodel geändert. Für nicht-kommerziellen Einsatz gibt es eine kostenlose Version: Nessus Essential, der bis 16 IPs scannen darf. Nessus basiert auf einem Client-Server-Modell. Der Nessus-Server nessusd ist für die Durchführung der eigentlichen Schwachstellentests verantwortlich. Der Nessus-Server wartet auf eingehende Verbindungen von Nessus-Clients, mit denen Endbenutzer bestimmte Scans konfigurieren und starten. Nessus-Clients müssen sich beim Server authentifizieren, bevor sie Scans starten können[2]. Die Architektur wird auf den Bild angezeigt 4.2.
Die Nessus-Software arbeitet etwas anders als vergleichbare Schwachstellenscanner.
Statt eine einzige, umfassende und ständig aktualisierte Datenbank vorzuhalten, unterstützt Nessus die Nessus Attack Scripting Language (NASL). Diese ermöglicht es den Sicherheits-Spezialisten eine einfache Sprache für die Beschreibung individueller Attacken zu benutzen. Der Nessus-Administrator fügt dann ganz einfach die NASL-Beschreibungen aller gewünschten Schadpotenziale ein und entwickelt so auf seine Bedürfnisse zugeschnittene Scans [Sch19].
Die Basis des Scanprozesses bei Nessus sind die *Plug-Ins*. Ein Plug-In wird in NASL geschrieben und besteht aus Schwachstelleninformation, allgemeinen Korrekturmaßnahmen und dem Algorithmus zum Prüfen der Existenz der Sicherheitsschwachstelle. Alle Plugs-Ins gehören zu Plug-In Familien. Es ist sehr praktisch bei der Scankonfiguration. Nessus erlaubt, die ganze Familie in den Scan einzufügen, ohne einzelnen Plug-Ins zu aussuchen.

z.B. die Plug-In Service Detection besteht aus 463 Plug-Ins. Es wäre aufwändig, eines nach dem anderen einzufügen. In dieser Menge von der Plug-Ins wird schnell die Übersicht verloren.

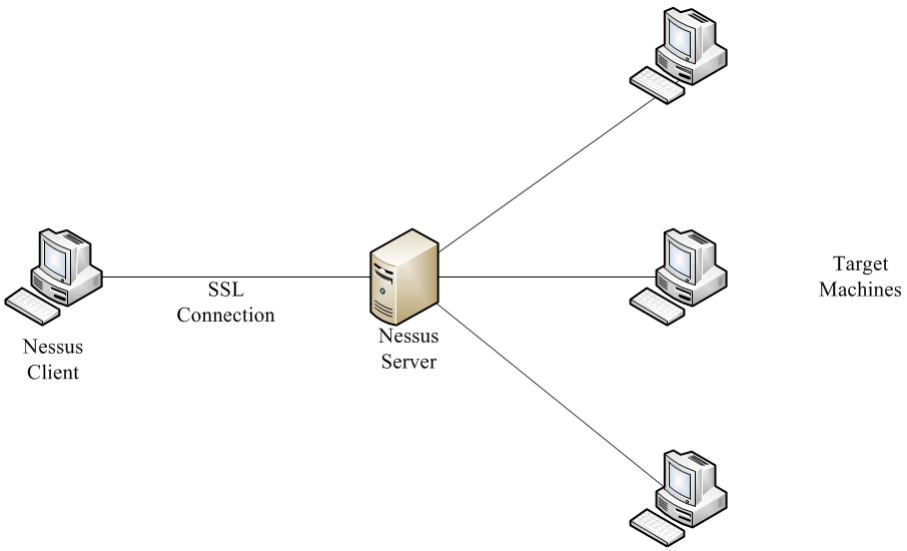

Abbildung 4.2: Teanable Nessus Arkitektur [nea14]

Nessus stützt sich auf die „Common Vulnerabilities und Exposures“ (CVE) Architektur, die das Cross-Linking kompatibler Sicherheits-Tools ermöglicht [Sch19].
Der Vulnerability Scanner Nessus ist eines der leistungsfähigsten und flexibelsten Tools zur Schwachstellenerkennung im Netzwerk. Seine Stärke liegt in der Vielzahl der erkannten Schwachstellen und in der Flexibilität der Nessus Attack Scripting Language [Sch19].

Für das Experiment wurde eine kostenlose Version Nessus Essentials benutzt.

OpenVAS

OpenVAS wurde von Nessus abgespalten, als Nessus 2005 zu einer proprietären Lizenz wechselte. Um weiterhin eine freie Version zu haben, wird OpenVAS seitdem auf Basis der letzten freien Version von Nessus weiterentwickelt. Version 1.0 erschien im Oktober 2007 [wik19].

Das Open Vulnerability Assessment System (OpenVAS) ist eine Kombination aus mehreren Diensten und Werkzeugen, die zusammen eine umfangreiche und mächtige Lösung für Schwachstellen-Scanning und Schwachstellen-Management darstellen[Bun].
Die ganze Architektur wird im Bild angezeigt 4.3.

Herzstück des OpenVAS ist der „OpenVAS Scanner“, der die Rolle des virtuellen Penetration Testers übernimmt und die eigentlichen Tests ausführt, indem er die bereitgestellten Werkzeuge, wie beispielsweise den Portscanner nmap, nutzt. Im gesamten OpenVAS wird verschlüsselt kommuniziert. Informationen darüber, wie ein Test auf etwaige Schwachstellen ausgeführt werden soll, liefert ein Feed, der die sogenannten Network Vulnerability Tests (NVTs) zur Verfügung stellt. Dieser Feed ist in einer kostenfreien Variante als Greenbone Community Feed (GCF) und in einer kommerziellen Variante als Greenbone Security Feed

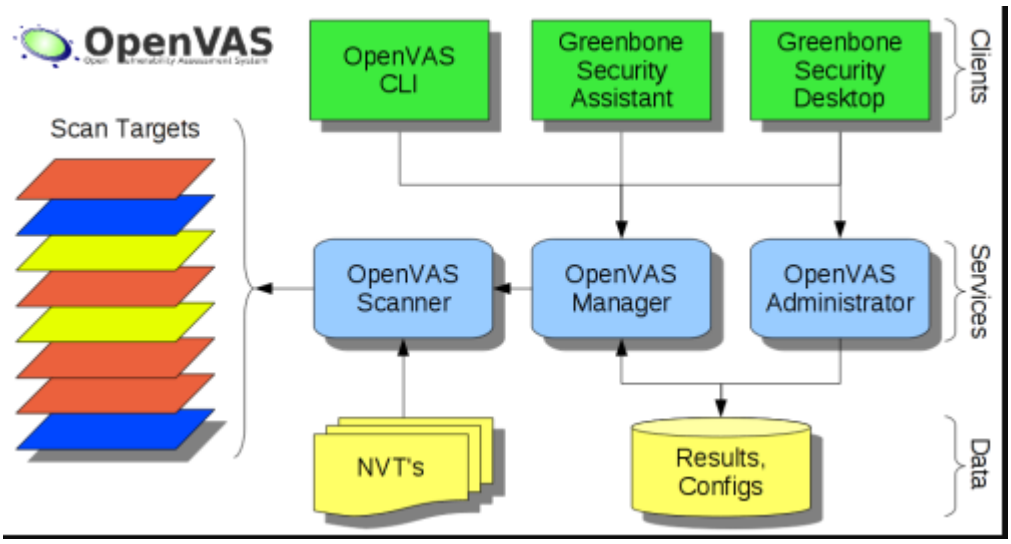

Abbildung 4.3: OpenVAS Struktur [Bun]

(GSF) erhältlich [Net18].

NVTs sind nicht anderes als Plug-Ins in Nessus. NVTs sind auch in der NASL Sprache geschrieben und werden mit der CVEs angebunden.

Den Hauptbestandteil des Systems bilden die NVTs, die Anweisungen für das Erkennen von Sicherheitslücken bereitstellen. Zu Beginn eines jeden Tests wird ein Profil des Systems erstellt. Anhand der erstellten Profile werden Tests ausgewählt, die für das System infrage kommen. Beispielsweise wird ein System nur dann auf fehlende Windows-Updates untersucht, wenn es sich um ein Windows-System handelt. Die Tests werden dann durch den OpenVAS-Scanner effizient anhand der erstellten Profile ausgeführt. Die Profile und Ergebnisse der Tests werden im Hintergrund durch den OpenVAS-Manager abgelegt und verwaltet [Net18].
Die Aktualisierung des NATS wird nicht automatisch durchführt. Man muss das manuell selbst machen oder Software installieren, die diese Funktion bereitstellt, z.B. Cron. Wie in Nessus werden alle Sicherheitslücken in verschiedene Risikostufen unterteilt (Severety Classes). Das hilft dem User Überblick über kritische Schwachstellen zu bekommen.

Für das Experiment wurde OpenVAS Verion 9 genommen.

Auf Basis der Studien [Ten18], die Tenable veröffentlicht hat, wurde folgende Vergleichstabelle von Nessus, OpenVAS und Rapid7 Insight erstellt 4.4.

4.3 Dienste

Unten wird angezeigt, welche Dienste im Experiment benutzt werden, um die Netzwerkschwachstellen Scanner zu evaluieren. Um mehr Information zu gewinnen, werden nicht

	Nessus	OpenVAS	Rapid7
CVE-Abdeckung	47.000+ CVEs - die meisten in der Branche	< 26,000 CVEs	< 38,000 CVEs
Scan-Genauigkeit	Die branchenweit niedrigste Falsch-Positiv-Rate	Nicht veröffentlicht	Nicht veröffentlicht; Kunden berichten von vielen Fehlalarmen
Vorgefertigte Scanvorlagen	Vorlagen für wichtige Sicherheitslücken (WannaCry, Spectre & Meltdown usw.), SCAP- und OVAL-Auditing und mehr	Keine vorgefertigten Vorlagen für WannaCry, Spectre & Meltdown usw.	Keine vorgefertigten Vorlagen für WannaCry, Spectre & Meltdown usw.
Live-Ergebnisse	Live Results identifiziert Schwachstellen mithilfe vorhandener Scandaten mit neuen Plug-Ins-Updates und sorgt so für Echtzeit-Transparenz	Nicht verfügbar	Kontinuierliche Überwachung
Risikobewertungen	verfügbar	verfügbar	Verfügbar

Abbildung 4.4: Vergleichstabelle von Nessus, OpenVAS und Rapid7 Insight

immer aktuelle Versionen der Dienste benutzt. Die Dienstversionen werden in der Tabelle angezeigt 4.5

1.Webserver: Apache HTTP
Apache ist einer der meistbenutzten Webserver. Der Webserver wird benötigt, um Webseiten auf dem Server per HTTP bzw. HTTPS Protokoll anzeigen zu können. Einzelne Benutzer bauen über ihren Browser eine Verbindung zum Server auf und fragen nach einer bestimmten Webseite. Der Webserver liefert daraufhin die gewünschte Webseite an den Browser zurück.

2. SMTP
Mailserver sind für einen geregelten E-Mail-Verkehr nötig. Die meist aus mehreren Modulen bestehende Software kümmert sich um das Senden, Empfangen und Verteilen von E-Mails an die entsprechenden Benutzer.

3. Active Directory Domain Webservice
Active Directory-Webservice ist ein Windows-Dienst, der die Remoteverwaltung aller lokalen Verzeichnisdienstinstanzen mithilfe von Webservice-Protokollen ermöglicht.

4. Datenbankserver: MySQL
Datenbankserver haben die Aufgabe, große Mengen von Daten effizient und permanent zu speichern. Ein Datenbankmanagementsystem sorgt dafür, dass benötigte Daten schnell aus der Datenbank herausgesucht werden und dem Benutzer übersichtlich angezeigt werden können.

5. Datenbankserver: Apache CouchDB
CouchDB ist auch ein Datenbankmanagementsystem. Es gehört zu der Familie NoSQL.

6. Microsoft Internet Information Services (IIS)
Diese Software ist in das Betriebssystem integriert und enthält Serverdienste um die wichtigsten Aufgaben eines Servers zu übernehmen. Zum Beispiel kann IIS als Webserver oder Dateiserver verwendet werden, ohne zusätzliche externe Dienste installieren zu müssen.

7. OpenSSH
Diese Applikation ermöglicht, eine gesicherte Verbindung zwischen zwei Computern aufzubauen. SSH verschlüsselt die Verbindung zwischen zwei Rechnern, sodass von einem Computer aus ein zweiter gesteuert werden kann.

8. DNS
DNS ist ein Netzwerkdienst, der Domänennamen in IP-Adressen konvertieren kann und umgekehrt.

9. FTP
Eines der grundlegenden Dateiübertragungsprotokolle, das zum Übertragen von Dateien zwischen Rechnern im Netzwerk entwickelt wurde. Mit FTP ist möglich, eine Verbindung zu FTP-Servern herstellen, den Inhalt von Verzeichnisse anzeigen und Dateien vom Server herunter zuladen oder auf den Server hochzuladen.

10. RDP Mit diesem Dienst wird ein Remote-Zugriff auf Server und Workstations realisiert. Die Kommunikation zwischen den Geräten erfolgt in Echtzeit über das Internet oder ein lokales Netzwerk.

In der Tabelle 4.5 werden wichtige Informationen über die genannten Dienste erläutert.

4.4 Zielsysteme

Wie es schon erwähnt wurde, wird das Experiment mit vier verschieden Windows Systemen durchgeführt. Es folgen kurz wichtige Informationen über die Betriebssysteme und Voraussetzungen für die Testversionen.

Windows Server 2016 Essential

Windows Server 2016 ist ein cloud-ready Betriebssystem, das derzeitige Workloads unterstützt und gleichzeitig neue Technologien einführt, die den Übergang zum Cloud-Computing erleichtern, sobald Sie dafür bereit sind [Mic19]. Die Essential Edition wurde für kleine Unternehmen mit Basisanforderungen an die IT entwickelt und gilt als Nachfolger von Windows Server 2012.
Testversion:180 Tage

Name	Protokoll	Versionsnummer	Standartportnummer
ftp	tcp	10.0.14393.0	21
OpenSSH	tcp	7.9	22
SMTP	tcp	10.0.143920	25
DNS	tcp /udp	10.0.14393.206	53
ISS	tcp	10.0.14393.	80
Apache	tcp	2.4	81
MySQL	tcp	5.7.26	3306
Microsoft Active Directory Web Server	tcp	10.0.14393.0	3268
RDP	tcp	10.0.14393.0	3389
Apache CouchDB	tcp	2.3.1	4369

Abbildung 4.5: Dienste

Windows 10 Enterprise

Windows 10 hat sieben verschiedene Variationen. Die Enterprise Edition basiert auf Windows 10 Pro, die für professionelle Anwender entwickelt wurde. Sowohl im Pro als auch Enterprise kommen zusätzliche Funktionen zur Verwaltung von Geräten und Apps sowie zur Einbindung in serverbasierte Windows-Netzwerke hinzu. Windows 10 Pro unterstützt das neue Windows Update for Business, um schnellere Sicherheits-Updates unabhängig von klassischen Patchdays zu bekommen [Imm16]. Der wichtige Unterschied zwischen Pro und Enterprise sind die Lizenzvoraussetzungen. Im technischen Sinn besteht kaum Unterschied.
Testversion: 90 Tagen

Windows 8.1

Version 8.1 ist der Nachfolger von Windows 8 und gilt als dessen verbesserte Version. Hier wird der erste Cloud-Dienst OneDrive präsentiert und die Suchmaschine Bing. Das Betriebssystem wird seit 2018 von Microsoft nicht mehr unterstützt.
Testversion: 90 Tagen

Windows 7 Professional

Windows 7 ist eines der beliebstesten Betriebssysteme unter Windows-Benutzern. Obwohl Windows 7 älter als Windows 8.1 ist, bietet Microsoft Support noch bis Ende 2020 an.
Testversion: 30 Tagen

5 Durchführung des Experiments

1. Vorbereitung der Evaluierungsumgebung

1.1. Installation und Konfiguration der Zielsysteme

Alle vier Betriebssysteme wurden auf den Laptop Pavilion HP installiert. Windows Server Essential 2016 wurde als primäres Betriebssystem auf dem Rechner aufgesetzt. Für die Installierung wurde eine ISO-Datei benutzt, die aus dem Windows Evaluation Center herunterladen wurde. Nach der Installation war es notwendig, sowohl die Netzwerkeinstellungen als auch Browsereinstellungen anzupassen, sodass man Zugang ins Netz bekommt, um das Herunterladen und den Scanprozess zu ermöglichen. Standardmäßig werden Firewalls eingesetzt, die für das Experiment nicht brauchbar sind. Um saubere Daten zu bekommen, werden bei allen Betriebssystemen, die im Experiment verwendet wurden, Firewalls abgeschaltet.

Als nächstes wurden die Dienste installiert.

IIS, SMTP, FTP, RDP, Web Server Active Directory wurden mit der Hilfe von Windows Server Manager installiert.
Windows Server Manager wird zum Anzeigen, Verwalten von Serverrollen benutzt und ermöglicht, Konfigurationsänderung durchzuführen. Mit dem Server-Manager können Administratoren lokale und Remote-Server verwalten, ohne physikalischen Zugriff zu haben.

Die weiteren Dienste mussten manuell installiert werden.

OpenSSH. Die Installationsdatei wurde auf dem Rechner gespeichert und der Pfad wurde im OpenSSH Ordner in „Environmetal Variables“ gespeichert. Danach wurde in Powershell folgende Kommandozeile eingegeben:

```
.\install-sshd.ps1
```

Bei der erfolgreichen Installation kommt die Meldung:`"sshd and ssh-agent services successfully installed"`. Um den Host Schlüssel zu generieren:

```
.\ssh-keygen.exe -A
```

MySQL wurde mit der Hilfe von MySQL Installer installiert. Dieses Tool ist für Windows Benutzer entwickelt. Für die Installation wurde es erforderlich, „Download Visual C++ Redistributable Packages for Visual Studio 2013“ auf den Server zu installieren.

Um die Installation von *Apache Webserver* durchführen zu können, wurde C++ Redistributable Visual Studio 2017 installiert. Danach wurde in der Konfigurationsdatei `httpd.conf`,

die sich im Installationsorder befindet, eine weitere Änderungen gemacht:

```
#Listen localhost:81
Listen 81
#Server Name localhost:81
```

Obwohl der Standardport für Webserver 80 ist, wurde hier 81 genommen. Grund dafür war, dass IIS den Port 80 belegte.
Um Installation zu beendet, wurde in cmd.exe ein weiterer Befehl eingegeben:

```
httpd -k install
```

Installationdatei von *CouchDB* ist eine EXE-Datei. Die Datei wurde ausgeführt und mit der Installation fortgefahren.
Alle oben genannten Dienste mussten aktiviert werden. Das passiert im Task Manager.

Um weitere Betriebssysteme als virtuelle Maschinen zu installieren, wurde Virtualisierungssoftware benötigt. Es wurde VirtualBox 6.0.8 Oracle auf dem Server installiert. Nach der erfolgreichen Installation wurden drei virtuelle Maschinen mit empfohlener Speicherkapazität angelegt und die Installationdatei zugewiesen:

Maschine 1: Windows 7 mit 2048 MB RAM und 32 GB Speicher
Maschine 2: Windows 10 mit 2048 MB RAM und 127 GB Speicher
Maschine 3: Windows 8.1. mit 2048 MB RAM und 40 GB Speicher

Als nächstes mussten die Einstellungen der Netzwerkanbindung bei den virtuellen Maschinen angepasst werden, sodass die für die externen Hosts (in unseren Fall für die Hosts, bei denen die Schwachstellenscanner eingerichtet wurden) erreichbar sein konnten. Standardmäßig werden bei den virtuellen Maschinen Netzwerkmodus NAT (Network Address Translation) eingestellt. Das bedeutet, dass der Gast auf das Netzwerk des Hosts Zugriff hat und auf das Internet, aber andersrum funktioniert die Kommunikation nicht.

Deswegen wurde hier der Netzwerkmodus: Netzwerkbrücke genommen. In diesem Modus bekommen virtuelle Maschinen direkten Zugriff auf das Hostnetzwerk. Die virtuelle Maschine wird direkt mit der virtuellen Netzwerkkarte an das Netzwerk des Hosts angeschlossen. Die Kommunikation funktioniert daher in beide Richtungen. Bei der Konfiguration kann ein Netzwerkadapter des Hosts gewählt werden [2]. VirtualBox fungiert als Switch. Der gesamte Netzwerkplan der Zielsysteme wird im Bild 5.1 angezeigt.

Nach erfolgreichen ping-Anfragen zwischen den Gast-Systemen und Host-Systemen wurde sichergestellt, dass die Evaluierungsumgebung zum Testen benutzt werden konnten.

1.2.Installation und Konfiguration der Netzwerkscanner

Die Scanner wurden so konfiguriert, dass sie gleiche Voraussetzungen haben. Die Scanprozesse bei den Tools übernahmen die gleichen Parameter:

Durchführung mit gleicher Menge an Portsnummern (1-6000)

Nutzung den gleichen Methoden bzw. Plug-Ins für Erkennung der Zielsysteme

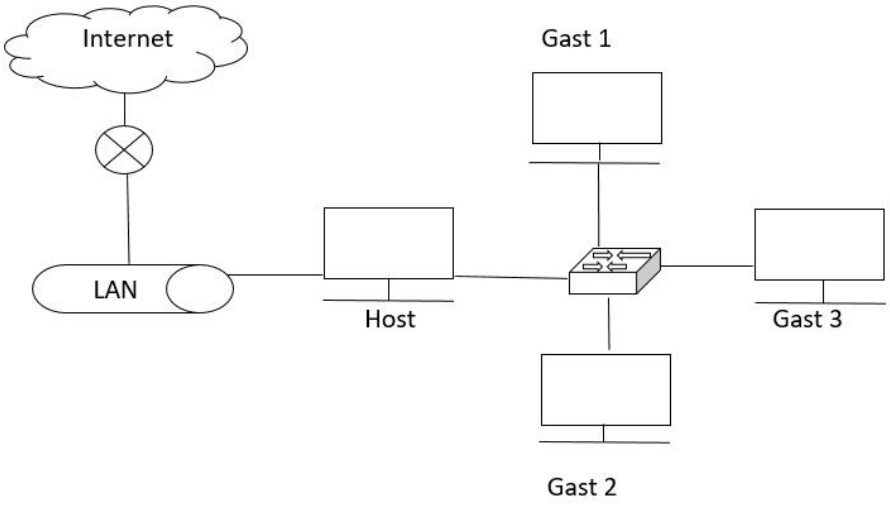

Abbildung 5.1: Netzwerkstruktur der Zielsystemen

Die Konfiguration der Scans wurde nur einmal am Anfang des Experiments durchgeführt und blieb für alle Scans unverändert.
Es wurde die Zeit protokolliert, die der Schwachstellenscanner für den Scann benötigt. Um eine grobe Zeitschätzung für den Scan zu erhalten, wurden die Tools einer nach dem anderen eingesetzt.

NMAP wurde auf Laptop Lenovo ideaPad mit Betriebssystem Windows 10 Home installiert. Für die Installation wurde eine EXE-Datei benutzt. Vorteil dafür ist, dass gleich alle Abhängigkeiten wie zum Beispiel Zenmap mitinstallieren werden können. Es dauert nicht lange. In wenigen Minute stand Nmap zur Verfügung. Für den Scan wurde die folgende Befehlszeile benutzt:

```
-nmap -A -sV -p 1-6000
```

Parameter `-A` steht für die Erkennung des Betriebssystems sowie dessen Version.
Parameter `-sV` ist der Parameter für Diensterkennung und dessen Version
Mit dem Parameter `-p` wurde der Intervall der Portsnummern eingegeben.

Die Installation weiterer Schwachstellenscanner auf den Lenovo ideaPad hat nicht funktioniert. Das Problem lag an nicht ausreichenden CPU und RAM. Deswegen wurde Nessus, Openvas und Repid7 Nexpose auf einem anderen Rechner installiert: Ein Acer Laptop mit Betriebssystem Ubuntu 16.04 und Hardware 8 GB RAM CPU, 4 2GHz cores.

Für die Installation von *Nessus Tenable Essentials* wurde die richtige Datei heruntergeladen und im Terminal wurde die Kommandozeile für Installation und Ausführen angegeben:

```
sudo dpkg -i NessusStandart_eh0muw.deb
sudo /etc/init.d/nessusd start
```

Um das Tools zu erreichen, muss man im Webbrowser den Link eingeben.

```
https://localhost:8834
```

Damit Nessus aktiviert wurde, musste man sich registrieren lassen und der Aktivierungscode kommt via Email. Nach der Aktivierung wurde die Installation der Plug-Ins durchgeführt, die ca. 30-40 Minuten gedauert hat. danach konnte man mit dem Tool arbeiten. Nessus hat vorbereitete Templates für den Scan, z. B “Host Discovery“, „System Discovery“, „Basic Network Scann“. Ein Template ist das vordefinierte Paket mit den Methoden, die für Scann verwendet werden können. Für das Experiment wurde kein relevantes Template ausgesucht. Alle verfügen über nicht benötige Plug-Ins und leider sind diese Templates unveränderbar für Benutzer. Deswegen wurde entschieden, das leere Template „Advanced“ zu nehmen und nur die relevanten Plug-Ins einzufügen:

```
Database: (inkl. SQL Server Version Detection)
General: (inkl. OS Identification, Post-scan OS Identification)
Service Datection
FTP(inkl. Core FTP Server Detection, vsftpd Detection)
Windows(inkl. Microsoft Remote Desktop Connection Installed)
Port-Scanner
```

Wenn man die Einstellungen für die PlugIns, Zielsysteme, Portnunmmern eingegeben hat, kann das Scan gespeichert werden und in der Zukunft wieder benutzt werden.

Installation *OpenVAS 9* wurde mit Hilfe von Terminal installiert.

```
add-apt-repository ppa:mrazavi/openvas
sudo apt update
sudo apt install sqlite3
sudo apt install openvas9
```

Oben genannte Zeilen fügen das Repository hinzu und installieren grundlegende Pakete des Tools. Um die Möglichkeit zu haben, Scanergebnisse als PDF-Datei zu speichern, wurde noch weitere Paketen installiert und der OpenVAS Prozess neu gestartet:

```
sudo apt install texlive-latex-extra --no-install-recommends
sudo apt-get install texlive-fonts-recommended
sudo service openvas-scanner restart
```

Nun wurde der OpenVAS-Manager neu gestartet und den Cache neu aufgebaut. Durch die Neuerstellung der Caches wird sichergestellt, dass die Pakete vollständig in den Manager geladen wurden.
Um das Webinterface zu benutzen, muss man im Webbrowser die Adresse eingeben. Standard Benutzereingabe sind `admin/admin` .

```
https://192.168.1.254:4000
```

Für die Konfiguration wurde zuerst ein neuer „Port Range“ eigerichtet. Danach wurde als „Target“ alle Zielsysteme mit neu erstellter Reihe von Ports gespeichert. Wie Nessus hat OpenVAS auch vordefinierte Templates, sogar fast mit identisch eingestellten Plug-Ins. Aber im Gegensatz zu Nessus kann man die Templates hier zuerst klonen und danach bearbeiten, Was sehr praktisch ist. Man muss dazu sagen, dass man fast von Allem in OpenVAS einen Klon realisieren kann (Port Range, Target, Tampelates, Task usw).
Für das Experiment wurde System Discovery als Grundlage für den Scann genommen. Aber es wurden weitere NVTs eingefügt, sodass Nessus und OpenVAS die gleichen Plug-Ins bekommen:

```
Datenbanken
FTP
```

Es wurde extra eingefügt, damit OpenVAS und Nessus den gleiche Methodensatz haben.
Nachdem die oben genannten Maßnahmen durgeführt wurden, wurde der Task mit unserem definierten Parameter erstellt. Dieser Task wurde während des Experiment mehrmals benutzt.

Rapid7 InsightVM. Wie bei der Installation der Nessus Tenable wurde verlangt, sich anzumelden. Im Email wurde der Aktivierungsschlüssel angegeben. Nachdem die Installation-Datei herunterladen wurde, wurde folgende Kommandozeile im Terminal eingegeben:

```
sudo md5sum -c Rapir7Setup-Linux64.bin.md5sum
sudo ./ Rapir7Setup-Linux64.bin
Um das Repaid7 Nexpose Prozess zu starten:
sudo system ctl start nexposeconsole
```

Um das Webinterface zu erreichen, muss man in Webbrowser die Adresse eingeben:

```
http://localhost:3780/starting.html
```

Bei der ersten Anmeldung hat der Initialisierungsprozess ungefähr 30 Minuten gedauert. Danach ist das Tool bereit zur Benutzung.
Wie bei den anderen Schwachstellenscanner wurde der Scan erstellt: man gibt das Zielsystem an und ein Port Intervall. Es stehen verschiedene Templates zur Verfügung. Analog zu OpenVAS ist es möglich, Templates zu klonen und bearbeiten.
Für das Experiment wurde ein Klon des Templates *„Discovery Scann“* erstellt und ein paar Änderungen eingefügt. Es wurde nicht nur Modus „Asset Discovery“, sondern auch „Vulne-

rability“ angeschaltet. Grund dafür war, dass die Datenbanken, FTP und Windows Dienste tiefer angeschaut werden konnten. Der Scan mit den Zielsystemen wurde gespeichert und kann beliebig häufig wiederholt werden.

2.Scans

2.1. Erster Scan: Standard

Da die Schwachstellenscanner viel Ressourcen brauchen, wurde entschieden, jeweils nur mit einem Scanner den Scan durchzuführen. Aber das Scan wurde gleichzeitig auf alle Betriebssystemen durchgeführt. Da NMAP auf einem anderen Rechner installiert wurde, wurde mit NMAP per einzelner IP Adresse gescannt.

2.2. Zweiter Scan: Änderung der Portnummern

Vor dem Scanprozess wurde von den Diensten verlangt, die Portnummer zu ändern.
Die Ports sowohl für IIS, als auch für FTP kann man im Internet Information Services (IIS) Manager ändern.
Die Porteinstellungen für MySQL Dienst können in MySQL Installer durchgeführt werden.
Die Portnummeränderung für Apache wird in dem Konfigurationsdatei `httpd.conf` gemacht. Dafür wird verlangt, folgende zwei Codezeilen zu ändern:

```
#Listen localhost:3389
Listen 3389
#Server Name localhost:3389
```

Für RPD wird die Portnummer im Registrierung-Editor eingegeben. In den Registrierungsunterschlüssel kann man beliebigen freien Port für RDP zuweisen:

```
HKEY_LOCAL_MACHINE
System CurrentControlSe Control TerminalServerWinStationsRDP-TcpPortNumber
```

Es ist empfohlen, den Rechner erneuert zu starten, sodass die Änderungen wirken. Der Zielsystemstand wird in der Tabelle 5.2 angegeben. Fett markierte Zeile zeigen die geänderten Dienste und Ports an.

2.3. Dritte Scan: Täuschungsversuch

Es wurde TCPOptimiser[tcp19] auf Windows Server 2016 installiert. TCPOptimiser ist eine Software für Optimierung und Einstellung der TCP/IP-Parameter. Das Tool ist für Windows Maschinen geeignet. Unten wird das TCPOptimiser Interface angezeigt5.3:

Es wurde das Flag *Time to Live(TTL)* auf den Wert 128 eingesetzt, was für Windows XP,7, Vista und Server 2008 relevant ist.

Name	Protokoll	Versionsnummer	Standartportnummer	Neu Portnummer
ftp	**tcp**	**10.0.14393.0**	**21**	**3306**
OpenSSH	tcp	7.9	22	22
SMTP	tcp	10.0.143920	25	25
DNS	tcp /udp	10.0.14393.206	53	53
ISS	**tcp**	**10.0.14393.**	**80**	**21**
Apache	**tcp**	**2.4**	**81**	**3389**
MySQL	**tcp**	**5.7.26**	**3306**	**6000**
Microsoft Active Directory Web Server	tcp	10.0.14393.0	3268	3268
RDP	**tcp**	**10.0.14393.0**	**3389**	**80**
Apache CouchDB	tcp	2.3.1	4369	4369

Abbildung 5.2: Scan2: Portnummer und Dienste

Bei der Windows Server 2016 war die Zahl 64 Standard. TCP Windows Size wird mit dem Registrierung-Editor eingesetzt:

`HKEY_LOCAL_MACHINE_SYSTEM_CurrentControlSet_Services_Tcpip_Parameters`

Als Wert wurde Linux (Kerne 2.4 und 2.6) *TCP Windows Größe* genommen: 5840. Alle Werten wurden aus der Tabelle genommen 5.4:

Für den Scan wurde das vordefinierte Template „System Discovery“(standardiesiertes Durchsuchen des Systems) genommen. Alle drei Netzwerkscanner haben ein solches Template. Nmap akzeptiert die gleiche Befehlzeile wie in Scan1 und Scan2. Der Scan wurde mit allen Schwachstellenscannern durchgeführt.

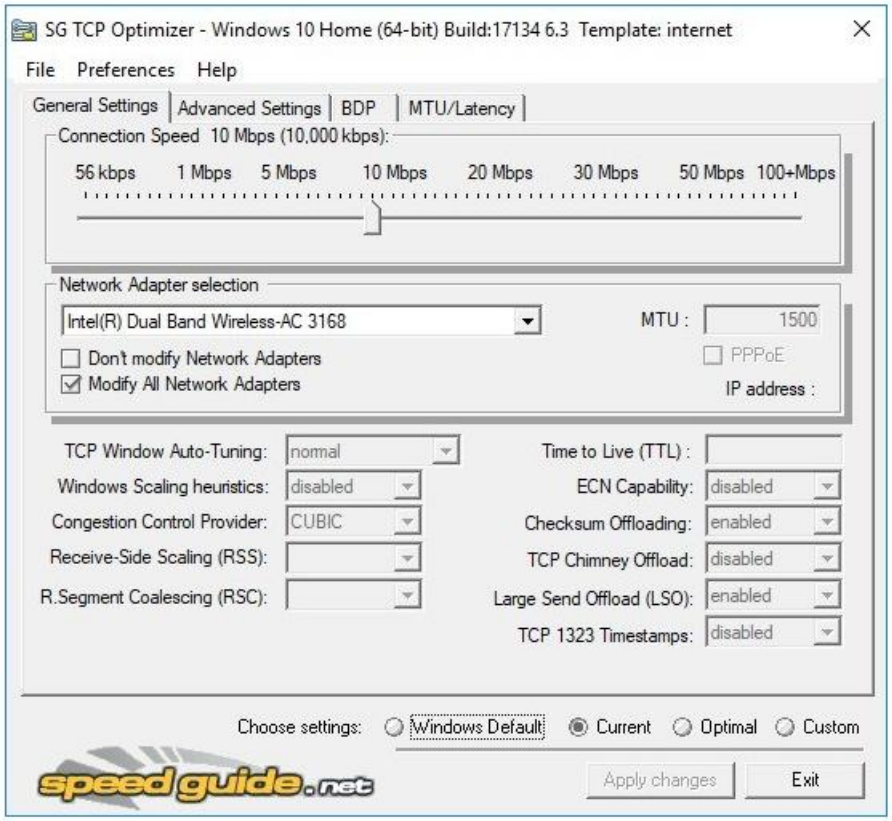

Abbildung 5.3: TCPOptimiser [tcp19]

Operating System (OS)	IP Initial TTL	TCP window size
Linux (kernel 2.4 and 2.6)	64	5840
Google's customized Linux	64	5720
FreeBSD	64	65535
Windows XP	128	65535
Windows 7, Vista and Server 2008	128	8192
Cisco Router (IOS 12.4)	255	4128

Abbildung 5.4: TCP Windows Größe und TTL [xen17]

6 Ergebnisse und ihre Interpretation

In diesem Kapitel werden zuerst die Ergebnisse mit der Interpretation von den drei durchgeführten Scanprozessen veröffentlicht. Die erworbenen und systematisierten Daten aus dem Experiment sind im Anhang A zur Verfügung. Sie sind in Diagrammen dargestellt.
Zum Schluss wird die allgemeine Schätzung der Schwachstellenscanner gemäß den zwei genannten Aspekten geäußert: Dienst- und Versionserkennung, Betriebssystemidentifikation.
Die durchschnittliche Zeit des Scanprozesses wurde hier auch kurz betrachtet

Scan 1.

Beim Windows Server Scan wurden fast alle Dienste von den Netzwerkscannern erkannt. Alle vier haben richtige Portnummern, Protokolle und Dienstname ermittelt.

Der Apache CouchDB wurde ausschließlich von NMAP identifiziert. Möglicherweise liegt das an den Betriebssystemen, da NMAP war der einzige ist, der auf einer Windows Maschine installiert wurde. Weshalb Nessus Apache CouchDB nicht identifizierte ist es unklar, da es im Vorfeld Tests mit Hilfe der gesamten Plug-In-Familie „Service Detection“ durchführen konnte. Hierbei führte ausschließlich **„Erlang Port Mapper Daemon Detection (id 62351)“** zur Identifizerung. Die gesamte Plug-Familie wurde auch in der Scanner Konfiguration des Experiments eingeführt, dennoch erfolgte keine Identifizierung.

Die Versionen von IIS und von manuell installierten Diensten (Apache, MySQLm OpenSSH) wurden von allen Scannern richtig erkannt. Die Schwachstellenscanner extrahieren in den meisten Fällen Informationen über die Versionen aus den Bannern der jeweiligen Dienste.
Als Beispiel wird SQL herangezogen. Die Ergebnisse von Versionserkennung: OpenVAS (Version wurde aus dem Banner extrahiert), Rapid7 (Version wurde in den zusätzlichen Informationen ohne benutzende Methode angegeben) und NMAP (keine Eingabe von Methode). Hingegen nutzt Nessus für die Bestimmung von Version gleichzeitig mehrere Plug-Ins (**10719 - MySQL Server Detection**, **45590 - Common Platform Enumeration (CPE)**.

Die Versionsnummer von Standarddiensten von Windows (rdp, ftp, adws, usw.) wurden von Netzwerkscannern nicht identifiziert.

Der Scan von leeren Windows Betriebssystemen lieferte sehr niedrige Ergebnisse der Versionserkennung, da ausschließlich der Dienst Microsoft-HTTPAPI 2.0 (Port 5357 Windows 8, Windows 10) identifiziert wurde.
Nessus war der einzige, der als Input „Web-Server“ ans Stelle Dienst Microsoft-HTTPAPI 2.0 eingegeben hat. Zudem zeigt es die besten Ergebnisse als andere Netzwerkscanner, da mehrere offene Ports und Dienste entdeckt wurden(siehe Diagramm 6.1). Die zweitbesten

Ergebnisse erzielte NMAP. OpenVAS und Rapid7 InsichtVM haben die gleiche Gesamtpunkte.

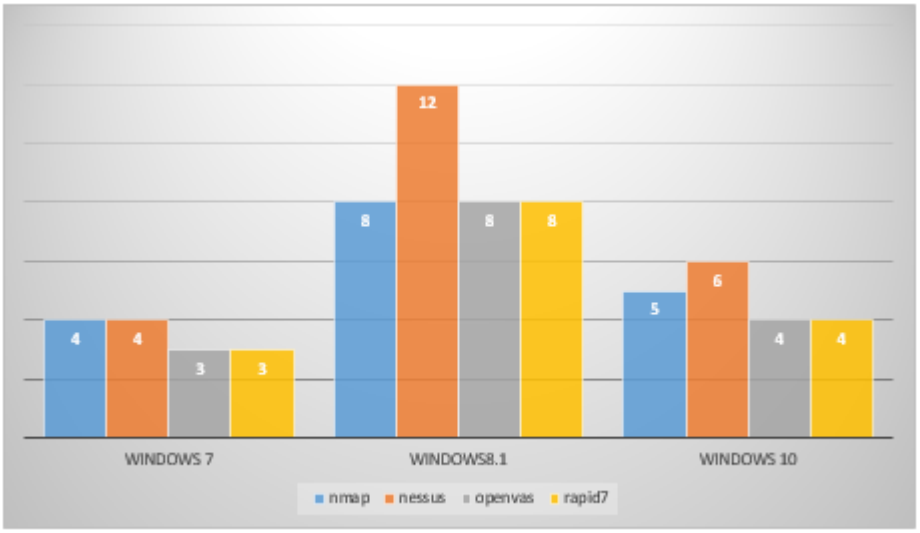

Abbildung 6.1: Ergebnisse der Scan1 und Scan2 bei Windows Server 2016

Die Betriebssystemidentifikation lieferte sehr gute Ergebnisse, da die Betriebssystemfamilie bei allen richtig bestimmt wurde.

Allerdings erkannte kein Scanner Windows 10 Enterprise.
Rapid7 hat das gleiche Ergebnis wie NMAP, da beide den Windows Server 2016 identifizierten. Nessus und OpenVAS hingegen konnten es nicht identifizieren. Das ist unklar, da bei den Tests im Vorfeld die Identifikation erfolgte. Bei Nessus erfolgte es mit Hilfe des Plug-In 10785 der Fam.„Windows“ (Port 445). Zudem erhielten Scanner Informationen über leere Desktop Windows durch das OS über SMB/Samba Banner 445/tcp und von DCE/RPC und MSRPC Services Enumeration 135/tcp bekommen.
Die gesamten Ergebnisse werden auf den Bild 6.2 dargestellt.

Scan 2.

Obwohl beim Scan 2 eine Portänderung vorgenommen wurde, sind die Ergebnisse von Scan 1 und 2 fast gleich. Lediglich zeigt sich ein Unterschied bei Nmap bei der Erkennung von offenen Port CouchDB, obwohl die Portnummer vom Dienst nicht geändert wurde. Dieser Unterschied entstand möglicherweise durch ein Verbindungsproblem.
Zudem konnten Nmap und Rapid 7 InsightVM RPD nicht den Port 80 finden.
Nessus und OpenVAS fanden es hingegen. Beide haben identische Ergebnisse nach Scan 1 und 2, was für hohe Vertraulichkeit bzgl. der Daten spricht. Die Ergebnisse von Scan 1 und Scan 2 wurden in Diagramm dargestellt 6.3.

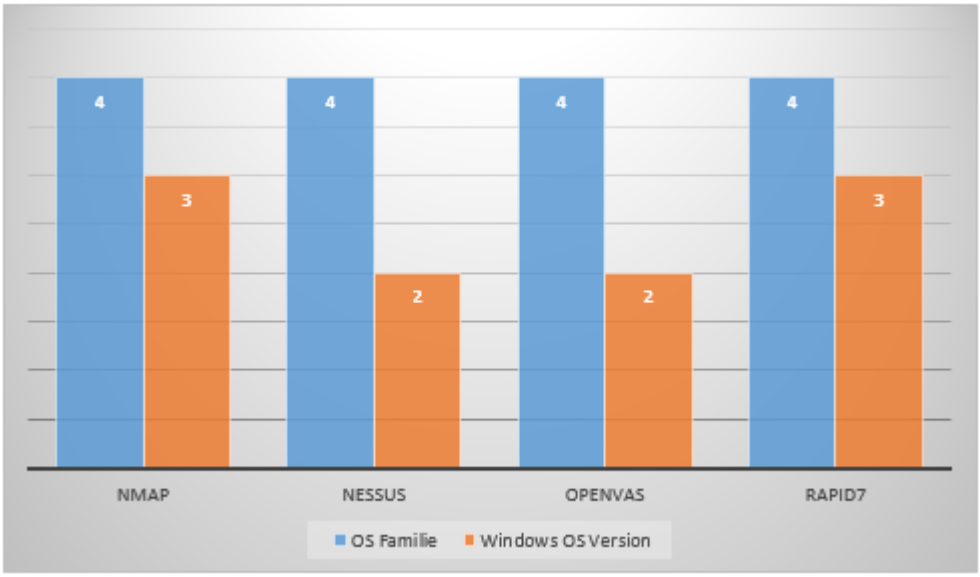

Abbildung 6.2: Betriebssystemidentifikation

Scan 3.

Die Betriebssystemidentifikation wurde wie im Scan 1 durchgeführt. Allerdings wurde ausschließlich Windows Server 2016 verwendet. Die Änderungen der TCP/IP-Parameters (TTL, tcp Windows Größe) zeigten keine Veränderungen der Ergebnisse. Deshalb sind sie identisch zu Scan 1.

Zeitmessungen

Für den Scan einer Ip-Adresse brauchte Rapid7 Insight VM 10 Minuten, wodurch es am längsten Zeit aufwandte. OpenVAS und Nessus hingegen brauchten bis zu 6 Minuten. Die Länge des Scanprozesses ist abhängig von der Anzahl der Ports, die gescannt werden müssen. Obwohl Nmap auf einem Rechner mit geringem Arbeitsspeicher eingesetzt wurde, brauchte es ca. 4 Minuten um das Ergebnis zu liefern.
Für die Initialisierung brauchten Rapid 7 Insight VM und Nessus. Die Nutzung des Tools nach dem Start dauerte ca. 20 bis 25 Minuten.Bei der OpenVAS war es max. 10 Minuten und bei der NMAP nur weringe Minuten.

Zusammenfassung der Scanner

Alle vier Scanner (Nmap, Nessus Tenable, OpenVAS und Rapid7 InsightVM) lieferten sehr gute Ergebnisse. Das Niveau der Ergebnisse unterscheidet sich max. um 10 %. Allerdings zeigte Nessus die besten Ergebnisse. Die Scanner untersuchten folgendes annähernd gleich gut: Dienst- und Versionserkennung sowie Betriebssystemidentifikation.

Zur Bestimmung von Diensten und Versionen wird von Nessus und OpenVAS mehre Me-

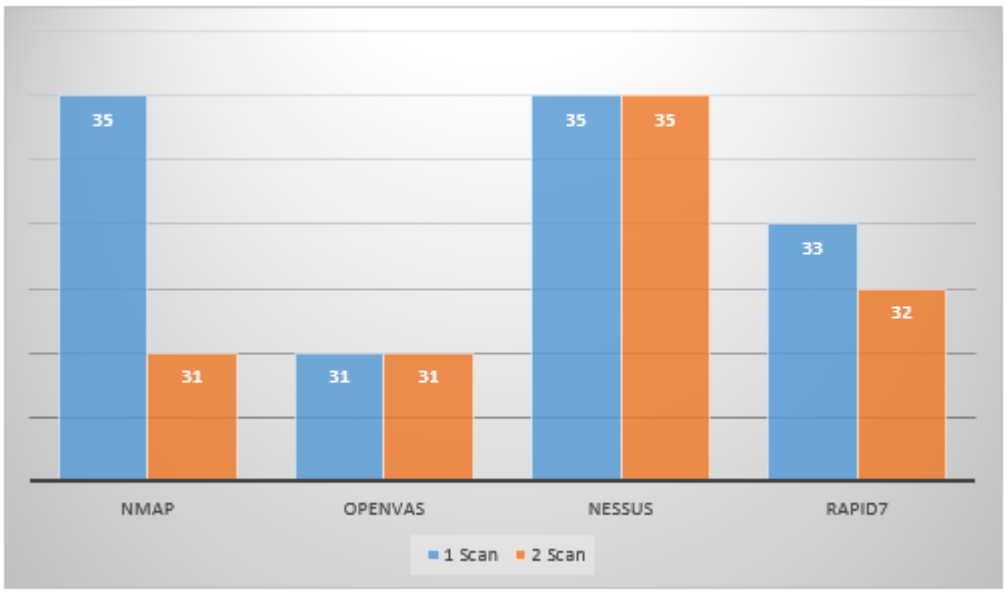

Abbildung 6.3: Ergebnisse der Scan1 und Scan2 bei Windows Server 2016

thoden bzw. Plug-Ins genutzt. Dadurch wird die Genauigkeit der Daten des Zielsystems gewährleistet. Sie lieferten bei der OS Identifikation gute Ergebnisse.

NMAP ist ein sehr effektives Tool, das die Ergebnisse über das System (Dienst, Dienstversion und OS) mit sehr hohe Zuverlässigkeit liefert.

Rapid 7 stürzte während des Experiments ein paar Mal ab und brauchte somit mehr Zeit als die anderen.

7 Zusammenfassung und Ausblick

In Rahmen der Arbeit wurde erforscht, wie gut NMAP und die Schwachstellenscanner bei der Diensterkennung, Versionserkennung und Betriebssystemerkennung des Windows Systems sind.
Um den Stand der Forschung des Problems zu erfahren, wurde eine Literaturrecherche durchgeführt. Hier geht es um die Ergebnisse der existierenden Studien, die sich mit der Funktionalität der Schwachstellenscanner sowie von NMAP beschäftigen. Ein großer Punkt war die Ausarbeitung der Theorie. Hier wurde genau angeschaut, welche Methoden bei der Diensterkennung heutzutage in Benutzung sind. Bei der Betriebssystemerkennung wurde die Arbeitsweise der Methode „des OS-Fingerprint“ erklärt.

Als nächsten Schritt wurde die Experimentdurchführung geplant. Hier wurde ein Scenario des Experiments ermittelt. Genau gesagt, wurden die Evaluierungsumgebung, Untersuchungsmaterialien und Durchführungsschritte vorgestellt.
Die Evaluierungsumgebung wurde aus folgenden Komponenten aufgebaut: NMAP und Netzwerkscanner, Zielsysteme und geeignete Dienste. Als Zielsysteme wurden die vier verschiedenen Windows Betriebssysteme aufgesetzt. Das zentrale Ziel wurde Windows Server 2016 Essential, das mit den ausgesuchten Diensten konfiguriert wurde.
Für die Untersuchung wurden die bekannte Schwachstellenscanner ausgesucht: Nessus Tenable Essential, OpenVAS 9 und Rapid7 InsightVM.
Das ganze Experiment bestand grob aus zwei Teilen: Vorbereitung und Scans. Bei der Vorbereitung erfolgte die Installation und Konfiguration sowohl der Zielsysteme, als auch von NMAP und der Schwachstellenscanner. Unter Konfiguration der Netzwerkscanner ist gemeint die Auswahl und Einstellung der geeigneten Methoden für die Diensterkennung.

Im Lauf des Experiments wurde drei Scans durchgeführt:

1. Scan: Dienste auf dem Windows Server 2016 hatten gewöhnliche Portnummern. Auch die weiteren Desktop Windows OS wurden ohne Änderungen gescannt. Hier wurde auch die OS Erkennung kontrolliert
2. Scan: Dienste dem Windows Server 2016 hatten ungewöhnliche Portnummern.
3. Scan: Änderungen in der TCP Einstellungen der Windows Server, um die Betriebssystemerkennung zu erschweren.

Nach der Analyse der bekommenden Daten wurden weitere Schlussfolgerungen getroffen:

Nessus Tenable Essentials, OpenVAS 9 und Rapid7 InsightVM hatten sehr gute Ergebnisse bei der Diensterkennung und OS Identifikation.
Versionserkennung der Standard Dienste von Windows (ftp, rdp, awds) wurde mit den Schwachstellenscannern nicht erkannt. Bei den solchen Diensten wie Apache, OpenSSH und

MySQL war das nicht der Fall.

Die Versionserkennung erfolgte meistens mit Hilfe der Banner Grabbing Methode.

Bei der Portnummeränderung der Dienste wurde festgestellt, dass der Diensterkennungsprozess von Portnummern unabhängig ist. Die Ergebnisse von Scan1 und Scan2 sind Beweis dafür.

Bei der Betriebssystemidentifikation hatten die Netzwerkscanner sehr gute Ergebnisse. Alle haben die richtige OS Familie erkannt. Nur in einzelen Fällen konnten die Schwachstellenscanner keine richtige Windows Version nennen. Die Änderung der Parameters TTL und TCP Windows Size hat kein positives Ergebnis gebracht. Die Ergebnisse für Windows Server 2016 haben sich nicht geändert. Das ermöglicht zu sagen: *Um das Betriebssystem zu identifizieren, setzen Scanner viel mehr Parameter und Methoden zur Untersuchung ein.*

Aus der Experiment kommen die weiter Aussagen über NMAP und drei weiteren Netzwerkscanner :

NMAP und die Schwachstellenscanner (Nessus, OpenVAS und Rapis7 InsichtVM) liefern fast identische Ergebnisse.

NMAP ist ein sehr effektives Tool, das die Ergebnisse über das System (Dienst, Dienstversion und OS) mit sehr hoher Zuverlässigkeit liefert.

Nessus und OpenVAS stehen große Anzahl von Plug-Ins bzw. NVTs zur Verfügung, wodurch mehr Informationen über das Zielsystem gewonnen werden. Die gleichzeitige Nutzung von mehreren Plug-Ins bei der Dienstidentifikation oder Versionsidentifikation führt zur präzisen und höheren Genauigkeit der Ergebnisse. Ein sehr wichtiger Vorteil von Nessus und OpenVAS sind die Scanberichte. Man kann nachvollziehen, welche Methoden für die Identifikation benutzt wurden.

Rapid7 InsightVM erzielte auch gute Ergebnisse. Die Ergebnisse von NMAP und Rapid7 sind sehr ähnlich, was vermutlich darauf zurückzuführen ist, dass Rapid im Hintergrund NMAP als Portscanner nutzt. Allerdings wird mehr Zeit für Scan mit Hilfe von Rapid7 gebraucht als mit anderen Scannern, da es während dem Experiment paar Mal abstürzte.

Bei der Betriebssystemidentifikation haben die Schwachstellenscanner keinen Vorteil gegenüber NMAP.

Meinung Meiner nach, muss ein passender Scanner ausgewählt werden, je nach Ergebnisse der jeweiligen Scanns von Zielsystemen:
Um eine allgemeine Übersicht und ziemlich präzise Ergebnisse zu erhalten, soll NMAP genutzt werden.
Für tiefere Untersuchungen des Systems würde ich vorschlagen, Nessus zu verwenden. Bei der OS Identifikation kann einer von vier Tools verwendet werden.

Im Rahmen einer zukünftigen Arbeit könnte man die Studie mit weiteren Schwachstellenscannern erweitern: Retina, Centraleyezer, Snyk, um bessere Tools für Diensterkennung zu finden. Es wird interessant, TCP/IP Implementierung bei Windows Server weiter zu ändern, sodass die Schwachstellenscanner das falsche Betriebssystemen als Ergebnis liefern.

Anhang A

In den weiteren Tabellen werden die Ergebnisse des Experiments angegeben (Scan1,Scan2,Scan3).

Es werden vier Aspekte der Diensterkennung betrachtet:

PN: Portnummer. Falls ein Port als offen erkannt wird, bekommt der Schwachstellenscanner 1 Punkt, ansonsten 0.

PP: Port Protokoll. Falls ein Port als offen erkannt wird, bekommt der Schwachstellenscanner 1 Punkt, ansonsten 0. Bei falschem Protokoll bekommt der Scanner ein 1 Punktabzug.

S: Dienst. Bei richtig erkanntem Dienst bekommt der Schwachstellenscanner 1 Punkt, ansonsten 0.

Bei falsch erkanntem Dienst bekommt der Scanner 1 Punktabzug

SV: Dienstversion. Bei richtig erkannter Dienstversion bekommt der Schwachstellenscanner 1 Punkt, ansonsten 0. Bei falscher Versionsausgabe bekommt der Scanner ein 1 Punktabzug.

Beispiel: OpenVAS hat Portnummer, Protokoll und Dienst von ftp richtig erkannt, aber die Dienstversion wurde nicht ermittelt. Für ftp bekommt OpenVAS 3 von 4 Punkten.

	Nmap 7.70				Nessus Tenable				OpenVAS 9				Rapid7 InsightVM			
	PN	PP	S	SV	PN	PP	S	SV	PN	PP	S	SV	PN	PP	S	SV
ftp	1	1	1	0	1	1	1	0	1	1	1	0	1	1	1	0

Scan1: Scan Windows Server 2016 Essential

	Nmap 7.70				Nessus Tenable				OpenVAS 9				Rapid7 InsightVM			
	PN	PP	S	SV	PN	PP	S	SV	PN	PP	S	SV	PN	PP	S	SV
ftp	1	1	1	0	1	1	1	0	1	1	1	0	1	1	1	0
OpenSSH	1	1	1	1	1	1	1	1	1	1	1	0	1	1	1	1
SMTP	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
DNS	1	1	0	0	1	1	1	1	1	1	1	0	1	1	1	0
ISS	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
Apache	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
MySQL	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
ADWS	1	1	1	0	1	1	1	0	1	1	1	0	1	1	0	0
RDP	1	1	1	0	1	1	1	0	1	1	1	0	1	1	1	0
Apache CouchDB	1	1	1	0	1	1	0	0	0	0	0	0	1	1	0	0

Scan1: Scan von leeren Windows-Desktop Betriebssystemen.

Hier werden alle Ports auf allen vier Windows Systemen angezeigt, die von Scannern als offen identifiziert wurden. Die Bedeutung der Abkürzungen und Punktevergebung sind gleich wie in Scan1

Windows 7 Proffesionl:192.168.178.59

	Nmap 7.70			Nessus Tenable			OpenVAS			Rapid7 InsightVM		
	pp	S	sV	pp	S	sV	pp	S	sV	pp	S	sV
135	1	1	0	1	0	0	1	0	0	0	0	0
137	0	0	0	0	0	0	0	0	0	1	0	0
139	1	0	0	1	0	0	1	0	0	1	0	0
445	1	0	0	1	0	0	1	0	0	1	0	0
1900	0	0	0	0	0	0	0	0	0	0	0	0
5040	0	0	0	0	0	0	0	0	0	0	0	0
5355	0	0	0	1	0	0	0	0	0	0	0	0
5357	0	0	0	0	0	0	0	0	0	0	0	0

Windows 10 Enterprise:192.168.178.63

	Nmap 7.70			Nessus Tenable			OpenVAS			Rapid7 InsightVM		
	pp	S	sV	pp	S	sV	pp	S	sV	pp	S	sV
135	1	0	0	1	0	0	1	0	0	0	0	0
137	0	0	0	1	0	0	0	0	0	1	0	0
139	1	0	0	1	0	0	1	0	0	0	0	0
445	1	0	0	1	0	0	1	0	0	1	0	0
1900	0	0	0	0	0	0	0	0	0	0	0	0
5040	1	0	0	1	0	0	0	0	0	1	0	0
5355	0	0	0	0	0	0	0	0	0	0	0	0
5357	1	0	0	1	0	0	1	0	0	1	0	0

Windows 8.1: 192.168.178.64

	Nmap 7.70			Nessus Tenable			OpenVAS			Rapid7 InsightVM		
	pp	S	sV	pp	S	sV	pp	S	sV	pp	S	sV
135	1	1	0	1	1	0	1	1	0	0	0	0
137	0	0	0	1	1	0	0	0	0	1	1	0
139	1	1	0	1	1	0	1	1	0	0	0	0
445	1	1	0	1	1	0	1	1	0	1	1	0
1900	0	0	0	0	0	0	0	0	0	1	1	0
5040	0	0	0	0	0	0	0	0	0	0	0	0
5355	0	0	0	1	1	0	0	0	0	0	0	0
5357	1	1	0	1	1	0	1	1	0	1	1	0

Scan2: Windows Server 2016 mit geänderten Dienstportnummern. Die geänderten Dienste sind fett markiert

	Nmap 7.70				Nessus Tenable				OpenVAS				Rapid7 InsightVM			
	PN	PP	S	sV	PN	PP	S	sV	PN	PP	S	sV	PN	PP	S	sV
ftp	**1**	**1**	**1**	**0**	**1**	**1**	**1**	**0**	**1**	**1**	**1**	**0**	**1**	**1**	**1**	**0**
OpenSSH	1	1	1	1	1	1	1	1	1	1	1	0	1	1	1	1
SMTP	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
DNS	1	1	1	0	1	1	1	1	1	1	1	1	1	1	1	0
ISS	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**
Apache	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
MySQL	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**
ADWS	1	1	1	0	1	1	1	0	1	1	1	0	1	1	0	0
RDP	**1**	**1**	**0**	**0**	**1**	**1**	**1**	**0**	**1**	**1**	**1**	**0**	**1**	**1**	**0**	**0**
Apache CouchDB	0	0	0	0	1	1	0	0	0	0	0	0	1	1	0	0

Scan 1,3: OS Identifikation Hier werden die Ergebnisse von Scan1 und Scan3 angezeigt.

WS 2016(1,3): Ergebnisse von OS Identifikation bei dem Windows Server 2016 in Scan1 und Scan2

	Nmap 7.70		Nessus Essential		OpenVAS 9		Rapid7 InsightVM	
	OS Familie	Version	OS Familie	Version	OS Familie	Version	OS Familie	Version
WS 2016 (1,3)	1	1	1	0	1	0	1	1
Windows 7	1	1	1	1	1	1	1	1
Windows 10	1	0	1	0	1	0	1	0
Windows 8.1	1	1	1	1	1	1	1	1

Abbildungsverzeichnis

Literaturverzeichnis

[AC04] Anton Chuvakin, Cyrus P.: *Kenne deinen Feind. Fortgeschrittene Sicherheitstechniken.* O Reilly, 2004

[ano11] anonym: *Banner Grabbing and OS Fingerprinting Techniques.* https://www.cvedetails.com/top-50-products.php?year=2018. Version: 2011

[ano19] anonym: *Service Name and Transport Protocol Port Number Registry.* https://www.iana.org/assignments/service-names-port-numbers/service-names-port-numbers.xml. Version: 2019

[Bun] Bundesamt für Sicherheit in der Informationstechnik (Hrsg.): *Open Vulnerability Assessment System (OpenVAS).* Bundesamt für Sicherheit in der Informationstechnik, https://www.bsi.bund.de/EN/Topics/Industry_CI/ICS/Tools/OpenVAS/OpenVAS_node.html

[DD08] Denise Dudek, Stephan Krause Lars Völker Martina Z. Christoph Sorge S. Christoph Sorge ; Faculty of Information Technology Rangsit University (Hrsg.): *Netzsicherheit und Hackerabwehr.Seminar WS07/08* . Universität Karlsruhe: Faculty of Information Technology Rangsit University , 3 2008. https://doc.tm.uka.de/tr/TM-2008-3.pdf

[Gla08] Gladbach, Bergisch ; FHDW (Hrsg.): *Verbesserung von OS- und Service-Fingerprinting mittels Fuzzing* . FHDW , 9 2008. https://arxiv.org/ftp/arxiv/papers/1403/1403.0439.pdf

[Imm16] Immler, Christian: *Windows 10: Unterschiede zwischen Home, Pro, Enterprise & Co.*, 6 2016. https://www.pc-magazin.de/ratgeber/windows-10-unterschiede-home-pro-enterprise-education-iot-core-mobile-3196312.html

[Lex12] Lexon e.U. (Hrsg.): *Backtrack 5.Effektive Sicherheitstests selbst durchführen.* Version 1.2. Holzing 52, A – 3252 Bergland: Lexon e.U., September 2012. http://lexon.at/wp-content/uploads/2016/03/Lexon_Backtrack-5.pdf

[Mic19] Microsoft (Hrsg.): *Windows Server 2016.* Microsoft, 2019. https://www.microsoft.com/de-de/licensing/product-licensing/windows-server-2016.aspx

[nea14] *Introduction to Nessus Vulnerability Scanner.* : *Introduction to Nessus Vulnerability Scanner*, 2 2014. https://www.speedguide.net/downloads.php

[Net18] Nethen, Nils von ; ORDIX AG (Hrsg.): *OpenVAS:Schwachstellenanalyse und -management.* ORDIX AG, 1 2018. https://www.ordix.de/ordix-news-archiv/1-2018/openvas-schwachstellenanalyse-und-management.html

[Nil06] Nilsson, Johan: *Vulnerability scanners*. Stockholm, Department of Computer and Systems Sciences Royal Institute of Technology, Diplomarbeit, 5 2006. – https://pdfs.semanticscholar.org/c2f1/1c1f68a523a01604e967dc1fe0a3f1ac2743.pdf

[nma15] nmap.org (Hrsg.): *News*. nmap.org, 2015. https://nmap.org/

[Pos81a] Postel, J.: INTERNET CONTROL MESSAGE PROTOCOL / RFC Editor. Version: 9 1981. https://tools.ietf.org/html/rfc792. RFC Editor, 9 1981 (792). – RFC. – 6 S.. – ISSN 2070–1721

[Pos81b] Postel, J.: INTERNET PROTOCOLL / Information Sciences Institute University of Southern California. Version: 9 1981. https://tools.ietf.org/html/rfc791. RFC Editor, 9 1981 (791). – RFC. – 11 S.. – ISSN 2070–1721

[R.17] R., PhD. K.: Comparative study of vulnerability scanning tools: Nessus vs Retina. In: *INTERNATIONAL SCIENTIFIC JOURNAL SSECURITY AND FUTURE"* 3 (2017), S. 69–71. – https://stumejournals.com/journals/confsec/2017/2/69/pdf

[Rapa] Rapid7 (Hrsg.): *insightVM FEATURES*. Rapid7, https://www.rapid7.com/products/insightvm/features/

[Rapb] Rapid7 (Hrsg.): *insightVM Live-Schwachstellen-Management und Endpoint-Analytik*. Rapid7, http://www.cetus-consulting.de/_data/InsightVM_Product_Brief_-_German-Deutsch.pdf

[Rapc] Rapid7 (Hrsg.): *RAPID7 INSIGHT PLATFORM SECURITY*. Rapid7, https://www.rapid7.com/globalassets/_pdfs/whitepaperguide/rapid7-platform-cloud-security-overview.pdf

[Rei15] Reibold, Holger ; Brain-Media.de (Hrsg.): *NMAP kompakt*. Brain-Media.de, September 2015. http://www.reibold.de/Leseproben/Nmap%20kompakt%20-%20Leseprobe.pdf

[RL81] Rekhter, Yakov ; Li, Tony: TRANSMISSION CONTROL PROTOCOL / RFC Editor. Version: 9 1981. https://tools.ietf.org/html/rfc793. RFC Editor, 9 1981 (793). – RFC. – 15 S.. – ISSN 2070–1721

[SBG14] Sheetal Bairwa, Bhawna M. ; Gajrani, Jyoti: VULNERABILITY SCANNERS: A PROACTIVE APPROACH TO ASSESS WEB APPLICATION SECURITY. In: *International Journal on Computational Sciences & Applications (IJCSA)* 12 (2014), 2, S. 113–124. – https://pdfs.semanticscholar.org/6e29/253a9bc2c1e3f96b48ea7fced9403c2de2e4.pdf

[SC12] Sanon Chimmanee, Kritsada Sriphaew Aniwat H. Thanyada Veeraprasit V. Thanyada Veeraprasit ; Faculty of Information Technology Rangsit University (Hrsg.): *A Performance Comparison of Vulnerability Detection between Netclarity Auditor and Open Source Nessus* . Muang, Pathum Thani 12000 THAILAND: Faculty of Information Technology Rangsit University , 2012. http://www.wseas.us/e-library/conferences/2012/Paris/CICOCOM/CICOCOM-47.pdf

[Sch19] Schmitz, Peter ; Vogel Communications Group (Hrsg.): *Nessus-Lehrgang – Teil 1.Höchstleistung beim Schwachstellen-Scan.* Vogel Communications Group, 2019. https://www.security-insider.de/hoechstleistung-beim-schwachstellen-scan-a-193337/

[tcp19] *SG TCP Optimizer.* : *SG TCP Optimizer*, 2019. https://www.speedguide.net/downloads.php

[Ten18] Tenable (Hrsg.): *COMPARE NESSUS WITH INDUSTRY VULNERABILITY ASSESSMENT SOLUTIONS.* Tenable, 8 2018. https://www.tenable.com/products/nessus/nessus-professional/competitive-comparison

[Wal16] Walonka, Christian Andreas W.: *Flow-Record-Fingerprinting Host-, Dienst- und Software-Klassifikation basierend auf Flow-Records.* MÜNCHEN, LMU, Diplomarbeit, 11 2016. – http://www.nm.ifi.lmu.de/pub/Diplomarbeiten/walo16/PDF-Version/walo16.pdf

[wik19] *OpenVAS.* : *OpenVAS*, 4 2019. https://de.wikipedia.org/wiki/OpenVAS

[Wit19a] WithoutAuthor: *OS Identification(Grundlagen)*, 2019. https://www.elektronik-kompendium.de/sites/net/2103071.html

[Wit19b] WithoutAuthor: *Port-Scanning(Grundlagen)*, 2019. https://www.elektronik-kompendium.de/sites/net/2103051.html

[xen17] *zennolab.* : *zennolab*, 1 2017. https://zennolab.com/discussion/threads/kak-izmenit-versiju-os-cherez-tcp-ip.33372/