universität wien

MASTERARBEIT | MASTER'S THESIS

Titel | Title

Ein OpenVAS-basierter Ansatz zur Identifikation und Behandlung von Schwachstellen in Serverinfrastrukturen unter Anwendung des BSI-IT-Grundschutz-Kompendiums

verfasst von | submitted by

Mathias Wegerer BSc

angestrebter akademischer Grad | in partial fulfilment of the requirements for the degree of

Master of Science (MSc)

Wien | Vienna, 2026

Studienkennzahl lt. Studienblatt | Degree programme code as it appears on the student record sheet:	UA 066 926
Studienrichtung lt. Studienblatt | Degree programme as it appears on the student record sheet:	Masterstudium Wirtschaftsinformatik
Betreut von | Supervisor:	Univ.-Prof. Dipl.-Ing. Dr. Dr. Gerald Quirchmayr

Zusammenfassung

Die Absicherung eines Unternehmensnetzwerks und deren Systeme ist häufig mit erheblichem Aufwand verbunden. Bevor ein System geschützt werden kann, müssen dessen Schwachstellen identifiziert werden. Zu diesem Zweck kommen Verfahren wie Vulnerability Scans oder Penetrationstests zum Einsatz, um Kenntnisse über potenzielle Sicherheitslücken im System zu gewinnen. Anschließend sind geeignete Gegenmaßnahmen erforderlich, um diese Schwachstellen zu beheben. Das vom BSI veröffentlichte IT-Grundschutz-Kompendium beschreibt einen Katalog mit konkreten Maßnahmen, mit denen identifizierte Schwachstellen geschlossen werden können. Darüber hinaus bietet die automatisierte Kombination von Vulnerability Scans mit dem IT-Grundschutz-Kompendium einen effizienten Ansatz, um den Aufwand zu reduzieren und gleichzeitig Komfort sowie Sicherheit für IT-Sicherheitsanalysten und Unternehmen zu erhöhen. Diese Arbeit setzt sich zum Ziel, einen Lösungsweg zu konzipieren, der es erleichtert, Anforderungen des IT-Grundschutz-Kompendiums durch den Einsatz des OpenVAS Scanners zu evaluieren. Ein adaptierters Life-Circle-Modell bietet dabei die Grundlage für die Umsetzung eines Prototypen, der in unterschiedlichen Szenarien eingesetzt wird. Dabei werden sowohl authentifizierte als auch nicht-authentifizierte Scans durchgeführt. Die gezielte Auswahl geeigneter Network Vulnerability Tests (NVTs) ermöglicht es, technische Indikatoren zu identifizieren, anhand derer Rückschlüsse auf den Status der Erfüllung einzelner Anforderungen von IT-Grundschutz-Bausteinen gezogen werden können. Zusätzlich werden Herausforderungen erläutert, die beim Einsatz des OpenVAS Scanners auftreten.

Abstract

Securing a companies network and its systems is often associated with great effort. Before a system can be protected, its vulernarbilities need to be identified in first place. To gain insights into the security status of a system, methods such as vulnerability scanning or penetration testing are used. Subsequently, appropriate counter measures are required to close the potential security leak. The IT-Grundschutz-Kompendium published by the German Federal Office for Information Security (BSI) describes a catalog that include a huge collection of concrete measures that can be used to eliminate identified vulnerabilities. Furthermore, the automated combination of vulnerability scans with the IT-Grundschutz-Kompendium provides an efficient approach to reduce effort while simultaneously increasing the security and convenience for IT Security analysts and companies. This thesis aims to design a solution that facilitates the evaluation of IT-Grundschutz requirements using the OpenVAS scanner. An adapted life cycle model serves as the foundation for the implementation of a prototype that is applied in different sce-

narios. The targeted selection of suitable Network Vulnerability Tests (NVTs) makes it possible to identify technical indicators that allow conclusions to be drawn regarding the fulfillment status of individual requirements of IT-Grundschutz modules. In addition, the thesis discusses challenges that arise when using the OpenVAS scanner.

Inhaltsverzeichnis

1 Einleitung

1.1 Hintergrund & Motivation

Mit der stetigen Weiterentwicklung in der IT und dem Ausbau der weltweiten Vernetzung entstehen in vielen Bereichen des Alltags neue Möglichkeiten durch die Informatik, aber damit auch neue Sicherheitsrisiken. Die meisten Cyber-Angriffe sind das Ergebnis von bekannten Schwachstellen[25]. Jeden Tag versuchen Hacker sich durch das Ausnutzen von Sicherheitslücken unerlaubt Zugriff zu sensiblen Daten zu verschaffen [21]. Daher sollten Schwachstellen erkannt und geschlossen werden, um herauszufinden, ob diese von Servern, Netzwerkgeräten oder anderen Systemen ausgehen. Verwundbare Systeme können von einer Vielzahl an unterschiedlichen Angriffe kompromitiert werden, wie zum Beispiel einem DDoS-Angriff, Man-in-the-Middle oder einer SQL-Injection-Attacke. Diese Schwächen können durch falsche Konfigurationen oder unzureichende Policies, zum Beispiel einer fehlenden Passwort-Policy, entstehen. Grundsätzlich besitzt jedes System, das Informationen oder Services anbietet, Schwachstellen. Durch Vulnerability Scanning können zu einem gewissen Teil misskonfigurierte Systeme oder auch schwache Passwörter aufgedeckt werden[21]. Ein beliebtes Tool, das diesen Prozess unterstützt, ist OpenVAS. Dadurch ist es möglich, sicherheitsrelevante Informationen im Netzwerk zu sammeln und durch verschiedene Prüfungen Angriffspunkte im System zu erkennen, um sie schließlich zu bereinigen [7]. Zusätzlich zu bekannten Methoden und Tools haben es sich Organisationen wie das Bundesamt für Sicherheit in der Informationstechnik (BSI) zur Aufgabe gemacht, die Gesellschaft mit Dienstleistungen in den Bereichen Information, Beratung, Entwicklung und Zertifizierung zu unterstützen. Umgesetzt werden diese Dienstleistungen zum Beispiel in Form von veröffentlichten Standards oder Handbüchern.

1.2 Ziel der Arbeit

Diese Masterarbeit setzt sich zum Ziel, Verwundbarkeiten von Unternehmensnetzwerken wie auch einzelner Systeme aufzuzeigen und Lösungen zu deren Behebung, basierend auf dem IT-Grundschutz-Kompendium des BSI, vorzuschlagen. Steht in den ersten beiden Kapiteln dieser Masterarbeit eher theoretisches Hintergrundwissen im Vordergrund, wird in Kapitel 3 der OpenVAS Scanner ausführlich erläutert. Anschließend folgt in Kapitel 4 die Beschreibung des BSI, insbesondere des IT-Grundschutz-Kompendiums. Hier sind vor allem die System-Bausteine relevant, von denen überwiegend die Bausteine “APP“ (Anwendungen), “NET“ (Netzwerke) und “SYS“ (IT-Systeme) für diese Arbeit verwendet werden. In Kapitel 5 wird der in [21] beschriebene Life-Cycle erweitert. Darüber hinaus werden die Implementierung der einzelnen Phasen sowie die als Proof-of-Concept verwendeten Bausteine des IT-Grundschutz-Kompendiums disku-

tiert. Dabei gilt es, zahlreiche Herausforderungen zu bewältigen. Hierzu zählen insbesondere das Crawlen bzw. Extrahieren der Informationen aus dem IT-Grundschutz-Kompendium, das im Wesentlichen nur in rudimentären Formaten vorliegt, sowie die korrekte Platzierung und Konfiguration des OpenVAS-Scanners. Nach erfolgreicher Installation des OpenVAS Scanners folgt das Mapping der extrahierten Informationen des IT-Grundschutz-Kompendiums mit dem Scanergebnis von OpenVAS durch individuell programmierte Schnittstellen. Als Ergebnis erhält der User einen detaillierten Report über gefundene Defizite und daraus resultierende umzusetzende Maßnahmen nach den Vorgaben des IT-Grundschutz-Kompendiums. Der Prototyp wird schließlich in einem Musternetzwerk evaluiert, mit verschiedenen Test-Cases überprüft und die Ergebnisse anschließend diskutiert. Der Benutzer kann über einen Webbrowser auf den Prototyp zugreifen und die erforderlichen Befehle und Aktionen ausführen.

2 State of the Art

2.1 Ziel des Kapitels

Im Kontext der Identifikation von Schwachstellen in IT-Systemen existiert eine Vielzahl an Verfahren und Werkzeugen, wie beispielsweise Vulnerability Scanning, Penetration Testing sowie Tools wie Nmap oder OpenVAS. Ziel dieses Kapitels ist es, einen strukturierten Überblick über diese Ansätze zu geben, deren Funktionsweise zu erläutern und zentrale Unterschiede herauszuarbeiten. Darüber hinaus werden relevante Konzepte und Technologien dargestellt, die als Grundlage für die Entwicklung des in dieser Arbeit vorgestellten Modells dienen. Die gewonnenen Erkenntnisse bilden somit die Basis für die Konzeption und Umsetzung des im weiteren Verlauf der Arbeit entwickelten Prototyps.

2.2 Verwundbarkeiten

[26] definiert eine Verwundbarkeit als eine Schwäche, die in jedem Netzwerk präsent ist. Die Bedrohungen entstehen nicht durch die Verwundbarkeit selbst, sondern durch die Personen, die diese Schwachstelle ausnutzen, um einen Vorteil daraus zu gewinnen. Des Weiteren wird die Verwundbarkeit in einem Computersystem von [29] beschrieben als eine Schwäche im System oder eine ungewollte Schwachstelle in einem Software Code, die einem Eindringling Zugriff zum System verschafft. Dabei weist [29] darauf hin, dass Systemsicherheit durchaus auch bei privater Nutzung eines Computers oder für kleinere Unternehmen von Relevanz ist. Eindringlinge finden die verschiedensten Motivationen, warum ein System für Sie interessant ist. Sei es nun für einen Identitätsdiebstahl, den Zugriff auf E-Mail-Konten oder der Erwerb von Geheimnissen, die Motivationspanne für solche Angreifer ist nahezu endlos. Dabei führt [26] den Begriff Erkundung ein, der als Term zu verstehen ist, indem gezielt nach Informationen über das Opfer gespäht wird. Dieser Begriff wird in passive und aktive Erkundung unterteilt. Als aktive Erkundung bezeichnet [26] die direkte Interaktion mit dem Client durch Tools wie dig, whois, traceroute oder nslookup. Für eine mögliche Implementierung für passive Erkundung nennen die Autoren Netcraft, eine Platfform, die jede Webseite virtuell verfolgt und Daten bietet, die für den Angreifer wertvoll sind. Um die Verwundbarkeiten in einem Netzwerk zu entdecken, werden Netzwerk-Verwundbarkeits-Scans ausgeführt. Das Scannen von Ports wird oft genutzt, um damit Services der Zielmaschine zu erforschen. Auch ein Angriff auf ein Network kann sowohl aktiv als auch passiv erfolgen. Dabei bezeichnet [26] eine passive Netzwerkattacke als eine Aktivität, bei der der Netzwerkverkehr überwacht wird, um unverschlüsselte, sensitive Informationen und Passwörter, ohne dass das attackierte System Kenntnis bekommt, zu erlangen. Im Gegensatz dazu besteht das Ziel eines aktiven Angriffs darin, ein System gezielt zu kompro-

mittieren. Als Beispiele für passive und aktive Attacken, nennt [26] Spoofing und Sniffing. [26] bezeichnet Spoofing als die Verschleierung eines Systems, wo der Angreifer aktiv ein System imitiert. Sniffing bezeichnet die Aktivität, bei der ein System, das nicht dem Zielsystem entspricht, liest Daten wie zum Beispiel E-Mails, Passwörter, vertrauliche Informationen und viele mehr. Ein häufig eingesetztes Tool für Sniffing ist Wireshark. Für Spoofing kann zum Beispiel ETTERCAP eingesetzt werden. Dieses Tool kann für Man-In-The-Middle Attacken eingesetzt werden. Verwundbare Netzwerke können nach [21] von einer Vielzahl an Angriffen kompromittiert werden, wie zum Beispiel:

- DDoS
- DNS Spoofing
- DHCP Snooping
- ARP Poisoning
- Man-in-the-Middle
- Smurf Attacks
- Buffer Overflow
- SQL Injections

Diese Angriffe können Virus, trojanische Pferde, Worms, Malware oder auch Root Kits beinhalten. Die Ursachen von Verwundbarkeiten spiegeln sich in einer Vielzahl an Möglichkeiten wider, wie zum Beispiel:

- *Fehlende Patches:*
  Sicherheitspatches sind Softwareteile, die in eine bestehende Software integriert werden, um diese sicherer zu gestalten. Softwareentwickler stellen diese von Zeit zu Zeit den Endusern zur Verfügung, sofern zum Beispiel durch Forschung neue Erkenntnisse offenbart wurden, die die Software robuster gegen Hacker schützen könnte.

- *Schwache Passwörter:*
  Ein wichtiges Kriterium, das oftmals vernachlässigt wird sind Passwörter. Viele Passwörter für zum Beispiel Datenbanksysteme, Router, Firewalls oder ähnlichem sind mit einem schwachen- oder Defaultpasswort konfiguriert. Diese können mit wenig Aufwand ausgeforscht und benutzt werden, was ein System damit unnötigen Sicherheitsrisiken aussetzt. Ein Standardzugang zu einem Router/Switch von diversen Herstellern wäre zum Beispiel der username “admin“ in Kombination mit dem Password “admin“. Würde ein solches Standardpasswort nicht geändert werden, wäre es einem Angreifer ein Leichtes Zugriff zum Netzwerk zu bekommen.

- *Falsch konfigurierte Firewall-Regeln:*
  Eine äußerst wichtige Maßnahme gegen nicht gewünschten Zugriff oder schädliche Aktivitäten stellt die Firewall, beziehungsweise die definierten Regeln dieser dar. Eine falsch

konfigurierte Firewall birgt ein weiteres Sicherheitsrisiko für Netzwerke, weil damit unautorisierter Zugriff auf das Netzwerk erfolgen könnte. OWASP definiert zum Beispiel Regeln, die bei der Firewall-Konfiguration beachtet werden sollten.

- *Mobile Geräte:*
  Der Einsatz mobiler Geräte birgt immer ein gewisses Sicherheitsrisiko, vor allem weil diese oft auch für den privaten Gebrauch eingesetzt werden. Dazu speichern die meisten Geräte Passwörter, Cookies oder können einen Virus enthalten, welcher ein Risiko für ein Unternehmen darstellt. Des Weiteren können mobile Geräte leicht gestohlen und missbraucht werden.

- *USB Geräte:*
  USB-Geräte können befüllt sein mit sensiblen Informationen, Passwörter oder auch Viren. Sie bergen demnach ein ähnliches Sicherheitsrisiko wie mobile Geräte, weil sie ebenfalls gestohlen beziehungsweise das Netzwerk mit schädlichen Programmen infiziert werden können.

2.3 Maßnahmen

Für den Schutz gegen aktive Erkundung empfiehlt [26] eine Firewall in Kombination mit einem Intrusion Detection System. Da für eine passive Erkundung keine direkte Interaktion mit dem Client notwendig ist, ist ein Schutz hierfür schwieriger zu bewerkstelligen. [25] beschreibt folgende Maßnahmen zur Erhöhung der Sicherheit.

1. *Entfernung nicht benötigter Services:*
   Jeder Service, der über einen Webserver zur Verfügung gestellt wird, öffnet einen Port auf diesem. Daher steigt die Anzahl von potentiellen Schwachstellen mit jedem geöffneten Port. Ein erlangter Nebeneffekt besteht darin, dass die Performance steigen wird, wenn unnötige Services entfernt werden.

2. *Fernzugriff:*
   Fernzugriff wird oftmals benötigt, um den Dienst von Servern ununterbrochen zu ermöglichen und aus der Ferne zu warten. Dies birgt ein erhöhtes Sicherheitsrisiko durch das Erlauben von Fernzugriffen. Um dieses Risiko einzuschränken, sollte der Zugriff auf bestimmte Accounts und IP-Adressen beschränkt sein.

3. *Wartung von Benutzeraccounts:*
   Nicht benötigte User sollten gleichermaßen wie nicht benötigte Services entfernt werden, um Risiken zu minimieren.

4. *Berechtigungen und Privilegien:*
   Berechtigungen für User sollten genau analysiert werden, damit auch nur die erteilt werden, die tatsächlich notwendig sind, um Arbeiten durchführen zu können und somit dem User keine zusätzlichen Privilegien zur Verfügung stehen.

5. *Serverüberwachung:*
   Eine Serverüberwachung inkludiert das Erstellen von Reports, sowohl von Systemen als auch Applikationen, um in Folge unübliche oder verdächtige Aktivitäten zu erkennen und dementsprechend zu handeln.

[25] weist darauf hin, dass regelmäßiges Testen zur Erhöhung der Sicherheit von großer Relevanz ist. Das Scannen eines Computernetzwerkes sowie das Testen von möglichen Verwundbarkeiten basiert auf Tools und Prozessen. Dabei sollten nach [25] diese Tools und Ressourcen in den Sicherheitsrichtlinien jedes Unternehmens integriert sein, deren Informationssysteme für den Erfolg des Geschäfts notwendig sind.

2.4 Vulnerability Scanning

[29] definiert Vulnerability Scanning als einen Prozess, der einen Computer verwendet, um bei einem anderen Computer oder auch im Netzwerk, Schwächen aufzudecken. Dies stellt im Bereich der Cybersecurity einen wichtigen Schritt dar, um die entdeckten Schwächen zu beheben. Durch die Verwendung eines Vulnerability Scanning Tools wird für gewöhnlich ein detaillierter Report erstellt, der die gefundenen Verwundbarkeiten nach Schweregrad reiht. Als Überbegriff mit dem Vulnerability Scanning existiert das Vulnerability-Assessment, das nach [21] als ein Prozess definiert wird, der die Sicherheitslücken in einer Kommunikationsinfrastruktur, Netzwerk oder einem Computer beschreibt, identifiziert und klassifiziert. Dies inkludiert systematische Messungen, um Sicherheitslücken zu bewerten und zu reihen. Durch diese Analyse erhält das Unternehmen einen Überblick über die aktuelle Sicherheitslage und den Level an Bedrohungen. [21] definiert folgende Schritte, die das Vulnerability Assessment effektiver gestaltet:

1. Klassifikation von Werten, Fähigkeiten und Ressourcen
2. Zuordnung von Werten und Signifikanzen zu diesen Ressourcen
3. Identifizierung von Verwundbarkeiten und potentiellen Gefährdungen jeder Ressource
4. Abschwächen und Eliminierung der größten Verwundbarkeiten für die wichtigsten Werte, Ressourcen und Fähigkeiten
5. Wiederholung der Schritte 1 – 4 in einem definierten Zeitintervall

Vulnerability-Assessment kann sowohl einzeln als auch in Kombination manueller und automatischer Scans durchgeführt werden. Für das Management des Vulnerability Assessment präsentiert [21] den Vulnerabilities Management Life Cycle (Abbildung 2.1)

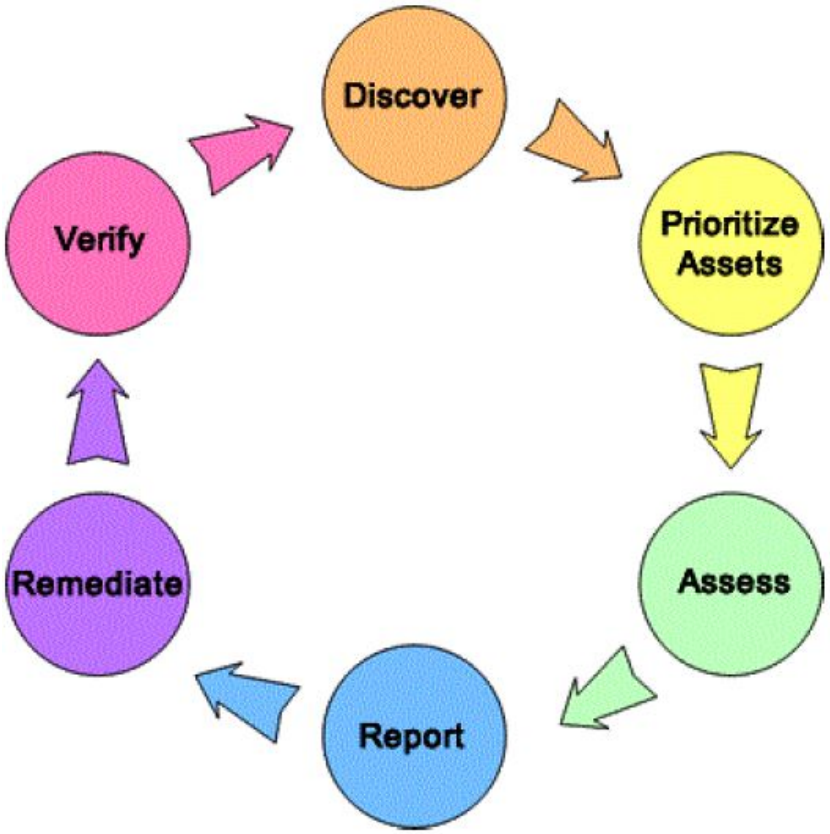

Abbildung 2.1: Vulnerabilities Management Life Cycle [21]

Dieser besteht aus den folgenden Phasen:

1. *Discover:*
   Dieser Schritt inkludiert das regelmäßige Auflisten aller Netzwerk-Komponenten, Betriebssystemen und Verwundbarkeiten der Systeme durch geeignete Tools.

2. *Prioritize:*
   Bezeichnet die Kategorisierung und Zuweisung von Werten zu diesen Gegenständen/Werte der Firma in Hinblick auf deren Priorität für das Unternehmen.

3. *Assess*
   Bezeichnet die Beurteilung der Werte des Unternehmens basierend auf Priorität, Risiken und Verwundbarkeiten zur Erstellung von Risikoprofilen

4. *Report:*
   Das Unternehmensrisiko soll in Bezug auf die Vorteile der Sicherheitsstrategie gemessen werden. Somit kann festgestellt werden welche Risiken es für das Unternehmen unbedingt zu beseitigen gibt.

5. *Remediate:*
   Hier werden die Maßnahmen durchgeführt, die notwendig sind, um die Verwundbarkeit zu beseitigen.

6. *Verify:*
   Nach der Schließung der Sicherheitslücke/n wird nun überprüft, zum Beispiel durch Sicherheitsaudits, ob diese auch tatsächlich beseitigt sind.

[20] definiert Vulnerability Assessment ähnlich und bezeichnet es als das Testen eines Computersystems, Netzwerkes oder einer Applikation zur Identifizierung, Messung und Reihung der Verwundbarkeiten innerhalb eines Systems. Dabei üben Hacker oft dieselbe Begutachtung aus wie Sicherheitsanalysen, um nach spezifischen Verwundbarkeiten zu suchen.

Bei der Durchführung eines Scans gilt es authentifizierte und nicht authentifizierte Scans zu unterscheiden. [19] beschreibt einen authentifizierten Scan als einen Scan, bei dem Authentifzierungsparameter, wie zum Beispiel Anmeldeinformationen eines Systems, für die Durchführung der Tätigkeit übergeben werden. Dies ermöglicht detailliertere und genauere Resultate. Außerdem verursachen authentifizierte Scans weniger Störungen auf dem untersuchten System, weil durch die verwendeten Benutzerinformationen Informationen einfacher abgerufen werden können. Nichtsdestotrotz weist [19] darauf hin, dass Anmeldeinformationen nicht immer gegeben sind und daher auch unauthentifizierte Scans durchgeführt werden müssen.

2.5 Penetration Testing

Penetration Testing, auch bekannt als Ethical Hacking [21], bezeichnet die Technik, die verwendet wird, um Verwundbarkeiten in einem System zu entdecken, bevor dies durch einen Angreifer geschieht. Dabei wird versucht wie ein Hacker zu agieren, wobei der Unterschied darin besteht, dass dies vom Eigentümer des Systems bewilligt und dieser Vorgang daher legal ausgeübt wird. Hierfür existieren unterschiedliche Methodologien:

- *Black Box Testing:*
  Bei dieser Methode wissen die Tester nichts über das interne Design oder der Struktur des Ziels. Dieses Testverfahren ähnelt der Methode eines Angreifers, der zum Zeitpunkt des Angriffs möglicherweise auch keine Informationen über das Unternehmensnetzwerk oder interne Strukturen besitzt. Hier werden zum Beispiel fehlende oder inkorrekte Funktionen/Interfaces getestet.

- *White Box Testing:*
  Hier besitzt der Tester vollständiges Wissen über alle internen Systeme und Funktionsweisen. Entwickler und Tester arbeiten bei dieser Methode für gewöhnlich zusammen, um etwaige Fehler zu finden. Dieses Wissen beinhaltet zum Beispiel Zugangsdaten, Prozeduren oder verwendete Protokolle.

- *Gray Box Testing:*
  Wie der Name bereits anzunehmen vermutet, liegt diese Methode zwischen den oben genannten Testverfahren. Hier besitzt der Tester einige interne Informationen über das System, muss aber durch eigene Recherche weitere Erkenntnisse über das zu infiltrierende System finden, um weiter vordringen zu können.

Des Weiteren erfolgt eine Unterscheidung nach verschiedenen Test-Typen:

- *Physisches Penetration Testing:*
  Beim Physisches Penetration Testing werden materielle Gegenstände getestet. Diese Gegenstände können zum Beispiel Server, Router, Switches oder auch Zutrittsbarrieren sein. Hier wird versucht die physischen Schwächen eines Systems auszunutzen, um unerlaubt Zugriff zu erhalten.

- *Virtuelles Penetration Testing:*
  Diese Form des Testens wird verwendet, um immaterielle Werte eines Unternehmens wie zum Beispiel Operating Systems, Web Server, Firewall, Datenbanken oder Ähnliches zu schützen. Die meisten Netzwerkangriffe werden durch Schwachstellen in diesem Bereich verursacht.

Ein weiteres Einteilungskriterium unterscheidet [21] nach den Bereichen des Penetration Testings wie zum Beispiel:

- Physical Penetration Testing
- Software Penetration Testing
- Database Penetration Testing
- Network Penetration Testing

Das Penetration Testing folgt grundsätzlich drei Phasen, die jeder Tester durchläuft, nämlich die Erkundung, Ausführung und Feststellung. Diese Phasen lassen sich in die folgenden Subphasen weiter unterteilen (Abbildung 2.2):

Abbildung 2.2: Penetration Testing Phasen [21]

1. Planning: Hier wird sowohl der Fokus des Tests bestimmt als auch die Rahmenbedingungen des Tests festgelegt

2. Reconnaissance: In dieser Phase werden so viele Informationen wie möglich über das Zielnetzwerk gesammelt. Diese gewonnenen Informationen können zum Beispiel IP Adressbereiche, offene Ports, Domainnamen, Operating System und andere Informationen sein.

3. Exploration: Die dritte Phase wird dazu genutzt, um das Netzwerk durch die gewonnenen Informationen aus der zweiten Phase näher zu erforschen. Diese Erforschungsphase soll dazu genutzt werden, Hosts, Services und Plattformen durch, zum Beispiel offene Ports, zu identifizieren.

4. Vulnerability Assessment: Vulnerability Assessment bezeichnet den Prozess, bei dem die Verwundbarkeiten eines Systems berechnet und sortiert werden.

5. Exploitation: Diese Phase bezeichnet die schwierigste Phase beim Penetration Testing, die sich mit dem Angriff auf das Zielnetzwerk beschäftigt. Hier wird versucht, die entdeckten Schwachstellen der letzten Phase zu verwenden, um das Netzwerk zu infiltrieren.

6. Reporting and Recommendation: Zum Schluss werden alle Aktivitäten, Erkenntnisse, verwendete Tools und Ähnliches in einem Dokument durch das Test-Team erfasst.

Darüber hinaus existieren zahlreiche Standards, die sich auf das Penetration Testing stützen. [21] listet hier unter anderem OWASP (Open Web Application Security Project) als auch den ISO Standard ISO/IEC27001:2005 (Information Security Management Systems) auf.

Die nachfolgende Tabelle aus [21] gibt einen kompakten Überblick über die Unterschiede zwischen Penetration Testing und Vulnerability Assessment.

Tabelle 2.1: Vulnerability Assessment vs. Penetration Testing [21]

	Vulnerability Assessment	Penetration Testing
Attributes	List-oriented	Goal-oriented
Type of Reports	Prioritized list of vulnerabilities categorized by criticality for remediation	Specific information of what data was compromised and vulnerabilities exploited
Purpose	Identify security vulnerabilities in system that may be exploited	Determine whether an application can withstand an intrusion attempt

Nmap

Es existieren zahlreiche Tools, die zur Durchführung von Sicherheitschecks verwendet werden. Im Folgenden werde ich Nmap vorstellen. Mit Nmap [26] steht ein weit verbreiteter Open-Source-Scanner zur Verfügung, der zur Analyse von Hosts und Diensten in Netzwerken eingesetzt wird. Ziel ist es, eine strukturierte Übersicht („Map“) der Netzwerkstruktur zu erstellen. Zu den zentralen Funktionen von Nmap zählen:

- Host-Erforschung
- Port-Scanning
  - TCP Connect Scan: Hier wird versucht, ein "Three-Way-Handshaking"mit jedem TCP-Port durchzuführen.
  - SYN Scan/Half-Open Scanning: Bezeichnet die Form des Scans, bei der IP Pakete generiert werden, wobei deren Antwort überwacht wird. Diese Form wird auch "Half-Open"bezeichnet, weil nie eine vollständige TCP Verbindung geöffnet wird.
  - NULL Scan: Sofern ein TCP Paket ohne einem RST Flag zu einem geschlossenen Port gesandt wurde, sollte als Antwort ein RST erhalten werden. Andernfalls erhält man keine Antwort. Somit kann bei Erhalt eines RST ein geschlossener Port und bei keiner Antwort ein offener Port angenommenen werden.
  - XMAS Scan: Identisch mit NULL Scan mit dem Unterschied, dass alle Flags bei ausgehenden TCP Paketen gesetzt werden.

  - FIN Scan: Geschlossene Ports antworten einem FIN Scan mit einem RST Paket, wohingegen offene Ports diesen Scan ignorieren.
  - UDP Port Scan: Hier wird ein UDP Paket auf jeden Port des Ziels gesandt. Jeder offene UDP Port sendet keine Antwort. Ein geschlossener Port sendet ein ICMP Paket mit einer "nicht erreichbarÄntwort.

- Versionserkennung
- OS Erkennung

2.6 Experimente

2.6.1 Verteiltes Vulnerability Assessment

Das Ziel von [20] ist es, ein Schema zu erzeugen, indem die Scan-Aufgaben eigenständig zu geringen Kosten durchgeführt werden. Diese eigenständigen Geräte werden durch ein Rasberry 2 Model B repräsentiert (Abbildung 2.3).

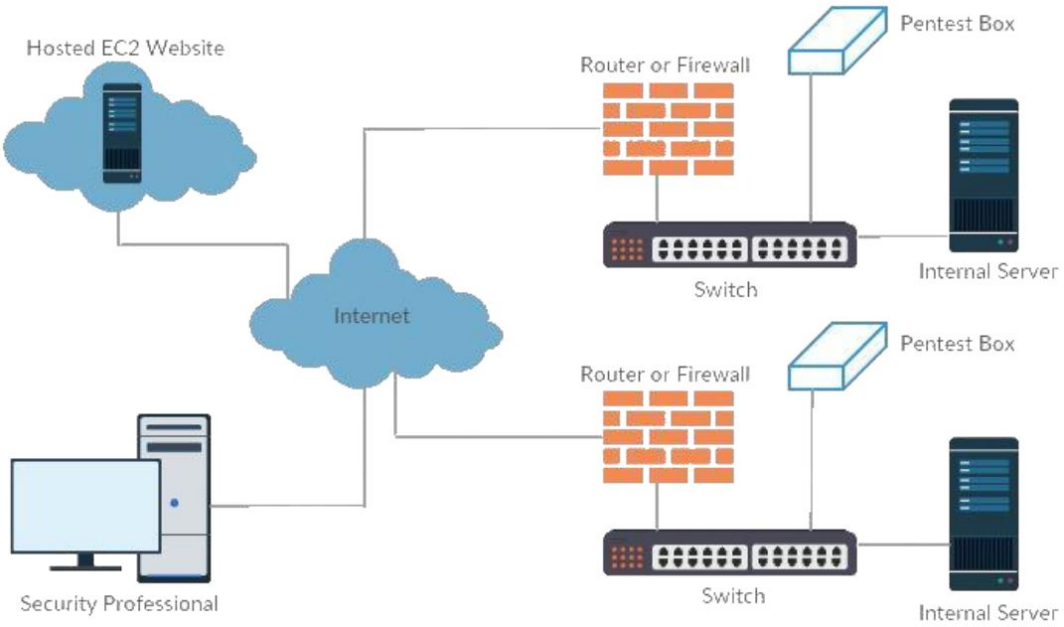

Abbildung 2.3: Netzwerkmodell [20]

Dabei platzierten die Autoren die Raspberry Pi's (Pentest Box) hinter einem Router/einer Firewall, wo diese Zugriff auf die Server haben, die gescannt werden sollen. Durch eine reverse SSH Verbindung verbindet sich die Pentest Box zu einer entfernten AWS EC2 Instanz. Auf dieser Instanz ist eine Anwendung gestartet, über welches Live Host Scans jeder Pentest Box durchgeführt werden kann. Anschließend kann der User über die Pentest Box einen Vulnerability Scan

mit OpenVAS bei jedem gefundenen Host durchführen. Nach dem Scan wird ein Report zurück an die Anwendung gesandt, die Informationen über die gefundenen Verwundbarkeiten enthält (Tabelle 2.2). Diese Phase, die OpenVAS startet, identifiziert verwundbare OS, Applikationen, Netzwerkservices und mögliche Backdoors. Der erhaltene Report wird anschließend in PDF und XML exportiert und in dem Open Source Penetration Testing Framework Metasploit importiert. Anschließend wird ein Python Skript angewendet, dass die Daten von der Metasploit Datenbank zur Dashboard Applikation auf dem AWS EC2 weiterleitet, um Vulnerability Assessment Reports zu erstellen.

IP Address	Operating System	Services	Vulns
192.168.200.174	Windows 2003	7	4
192.168.200.52	Windows XP	9	12
192.168.200.244	Linux	2	0

Tabelle 2.2: Übersicht identifizierter Systeme und Schwachstellen aus [20]

[20] weist darauf hin, dass sie Wochen benötigt haben, um eine lauffähige Version von OpenVAS auf dem Raspberry Pi funktionsfähig zu bekommen. Zusätzlich mit Hilfe von implementierten Python Skripts war es möglich, dass sich die Pentest Boxes automatisch bei der EC2 Anwendung registrieren.

2.6.2 Nessus und OpenVAS Test

[28] testet die beiden Vulnerability Scanner Nessus und OpenVAS in einem simulierten Netzwerk (Abbildung 2.4). Dabei nutzten die Autoren vier verschiedene Konfigurationsformen und verglichen diese:

- Scans ohne angemeldet zu sein mit/ohne Safe Check Option
- Scans mit Benutzeranmeldung mit/ohne Safe Check Option

Sofern Safe Check aktiviert ist, wird das Netzwerk nicht so aggressiv angegriffen, um die Gefahr zu vermeiden, dass möglicherweise durch den Scan ein Fehler im Netzwerk ausgelöst oder ein Service gestoppt werden könnte.

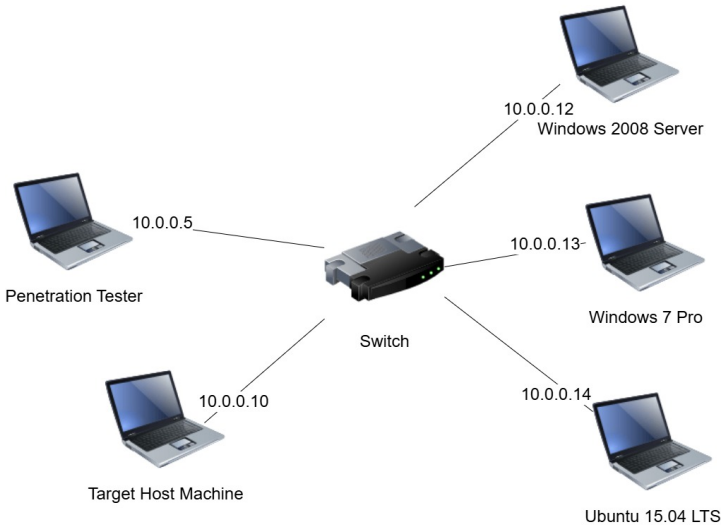

Abbildung 2.4: Netzwerk Simulationsumgebung [28]

Diese Scans wurden auf die Hosts im simulierten Netzwerk durchgeführt. Dabei verglichen sie die Resultate der beiden Scanner basierend auf den Common Vulnerability and Exposure (CVE) Kennzahlen, die vom MITRE Unternehmen entwickelt wurden. Diese Kennzahlen dienen auch als Basis für die U.S. National Vulnerability Database (NVD), die durch das National Insitute of Standards and Technology (NIST) bereitgestellt wird. Diese Datenbank bezieht sich auf verschiedene Informationen und Punkte für jede überwachte Softwareverwundbarkeit mit einem entsprechenden Schweregrad der Schwachstelle, basierend auf dem Common Vulnerability Scoring System (CVSS).

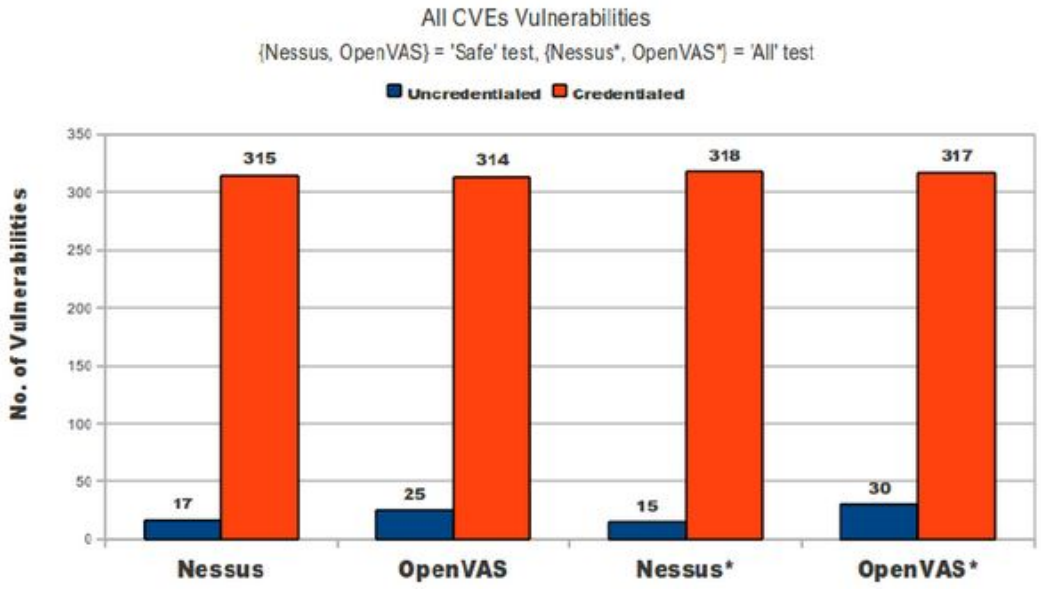

Abbildung 2.5: Nessus Vs. OpenVAS erkannte Schwachstellen [28]

Die Abbildung 2.5 zeigt die Ergebnisse, die die Autoren feststellten. Dabei entspricht der blaue Balken den Scan ohne entsprechender Benutzeranmeldung und die Gruppe, die mit einem Stern gekennzeichnet ist, die Scans, die ohne Safe Check Option durchgeführt wurden. [28] zeigt, dass ein Scan, bei dem Anmeldeinformationen benutzt werden, effektiver in der Erkennung von Schwachstellen ist.

Dieses Ergebnis wird auch durch die Autoren von [19], die verschiedene Scanner untersucht haben, bestätigt. Sie zeigen, dass ein Scanner mit Anmeldeinformationen genauere Ergebnisse sowohl hinsichtlich möglicher Gegenmaßnahmen als auch der Schwachstellen-Erkennungsrate liefert. In einem Fall wurde diese Rate von 2% auf 45% angehoben, wobei in [19] darauf hingewiesen wird, dass dies keinesfalls eine schlechte Rate sei. Denn Angreifer versuchen oft ohne Anmeldeinformationen Zugriff auf das System zu erhalten und dieser Extremfall von [19] zeigt, dass ohne Anmeldeinformationen Schwachstellen viel schwieriger zu entdecken sind.

[29] verwendet das Open-Source Tool OpenVAS für die Durchführung eines Vulnerability Scans. Dabei wird zunächst das Nmap Tool verwendet, um das Netzwerk zu scannen. Anschließend wird OpenVAS für das Scannen der gefundenen Hosts eingesetzt. [29] weist darauf hin, dass sich die Installation und Konfiguration von OpenVAS oft als komplex herausstellt, die Ergebnisse des Scans aber beeindruckende Outcomes liefern und als echte Alternative zu kommerziell erhältlichen Tools wie Nessus gelten.

2.7 Einordnung und Erkenntnisse

Die Analyse des aktuellen Stands der Technik zeigt, dass die Durchführung von Schwachstellenanalysen mit verschiedenen Herausforderungen verbunden ist, insbesondere im Hinblick auf die Konfiguration und den zielgerichteten Einsatz geeigneter Scan-Tools. Darüber hinaus wird deutlich, dass Vulnerability Scanning typischerweise als Bestandteil eines umfassenderen Prozesses betrachtet werden muss. In der Literatur wird es häufig als Subphase des Penetration Testings eingeordnet, welches als übergeordneter Ansatz mehrere aufeinander aufbauende Phasen umfasst. Für den weiteren Aufbau dieser Arbeit wird daher das Vulnerability Scanning in Kombination mit dem IT-Grundschutz-Kompendium betrachtet. Als konzeptionelle Grundlage dient dabei der Penetration-Testing-Life-Cycle, der im Rahmen dieser Arbeit angepasst und erweitert wird, um den spezifischen Anforderungen des entwickelten Modells gerecht zu werden. Die in [20] dargestellten Ergebnisse zeigen zudem, dass OpenVAS geeignete Exportmechanismen bereitstellt, die eine Weiterverarbeitung der Scan-Ergebnisse in externen Anwendungen ermöglichen. Diese Eigenschaft stellt eine wesentliche Voraussetzung für die im Rahmen dieser Arbeit angestrebte Verknüpfung von Scan-Ergebnissen mit Anforderungen des IT-Grundschutz-Kompendiums dar. Eine detaillierte Beschreibung von OpenVAS erfolgt in Kapitel 3.

3 Einsatz von OpenVAS im Vulnerability Scanning

OpenVAS beschreibt ein Open-Source Framework, welches aus verschiedenen Tools besteht und als mächtiges Werkzeug sowohl für Vulnerability Scanning als auch Vulnerability Management gilt [29]. Aufgrund seiner einfachen Bedienung und seinem umfangreichen Funktionsumfang wird dieses Tool durch das Bundesamt für Sicherheit in der Informationstechnik (BSI) beworben [7]. Vermutlich auch deswegen, weil deutsche Entwickler maßgeblich an der Entwicklung von OpenVAS beteiligt sind [27]. Die Software sammelt sicherheitsrelevante Informationen und klassifiziert diese in verschiedene Klassen. Dabei können verschiedene Konfigurationen vorgenommen werden, um die Prüftiefe von OpenVAS zu bestimmen. Dadurch kann der Sicherheitsstatus im Unternehmensnetzwerk festgestellt und ein Überblick über notwendige Maßnahmen gewonnen werden [7]. Darüber hinaus existieren zahlreiche Möglichkeiten, um die resultierenden Protokolle zu verwalten, auszuwerten oder zu exportieren. [12] gibt an, dass der Scanner täglich aktualisiert wird und mehr als 120.000 Tests umfasst. Die folgenden Kapitel sollen eine detaillierte Einsicht zu OpenVAS lieferen.

3.1 Historische Entwicklung von OpenVAS

Entwickler des Vulnerability Scanners Nessus entschieden sich 2005 dazu das Projekt unter einer Open-Source Lizenz fortzuführen, während Nessus fortan als proprietäre Software galt [12]. Nach den weniger aktiven Jahren 2006 und 2007 wurde im Jahr 2008 das Unternehmen Greenbone Networks GmbH gegründet, die sich seitdem um das Projekt OpenVAS kümmert. 2023 wurde das Unternehmen in eine AG umgewandelt [4]. Greenbone AG bietet unterschiedlichste Lösungen im Bereich IT Sicherheitsmanagement an:

1. Enterprise Appliances: Es werden sowohl Hardware- als auch virtuelle Appliances, bestehend aus dem Greenbone Operating System (GOS) und einem Scanner mit Weboberfläche, vom Unternehmen angeboten. Dabei werden die Schwachstellen aus dem Enterprise Datenfeed geschöpft und beinhalten dadurch wesentlich mehr potentielle Sicherheitsprobleme als in der Community Edition.

2. Cloud Service: Über den Cloud Service können IP Adressen gescannt und auf Sicherheitslücken untersucht werden.

3. Fortführung des Open Source Projekts um eine transparente Sicherheitslösung zu wahren (Community Edition)

[2] Mit der Unterstützung zweier weiterer Unternehmen, nämlich Secpod aus Indien sowie Security Space aus Kanada, wurde das Projekt weiterentwickelt und der Feed-Content wuchs rasant. Schließlich wurden 2009 die ersten Module hinzugefügt, um eine Vulnerability Management Solution zu erstellen. Dabei wurde das Web-Interface eigens entworfen, während zur selben Zeit der OpenVAS Scanner seine Kompatibilität zu vorherigen Versionen verloren hat. Im Frühjahr 2010 wurde schließlich der erste "Greenbone Security Manager" auf den Markt gebracht. In den darauf folgenden Jahren bis 2016 wurden an der kommerziellen Version und der Open-Source Version einige Verbesserungen und Weiterentwicklungen durchgeführt. Dabei wurden sie stetig vom Bundesamt für Sicherheit in der Informationstechnik unterstützt. Im Jahr 2017 hatte der Code des OpenVAS Framework bereits einige hunderttausend Codezeilen und es verging kaum ein Tag an dem nicht neuer Code hinzugefügt wurde. Zu jener Zeit war die 9.Version des Frameworks veröffentlicht. Weiters ist im Jahr 2017 die treibende Kraft hinter OpenVAS sichtbar geworden, nämlich Greenbone. Dadurch wurde der Name von "OpenVAS Framework" in "Greenbone Vulnerability Management" geändert, wo der OpenVAS Scanner eines von vielen Modulen darstellt. Dennoch verblieben die Module Open-Source. Eine zweite Änderung, die im Jahr 2017 stattfand, war die Änderung bezüglich des öffentlichen Feeds Services. Dieser wurde in "Greenbone Community Feed" umbenannt, um in von anderen OpenVAS basierten Produkten zu unterscheiden [3]. 2018 wurde der Transfer auf GitHub abgeschlossen. Anfang des Jahres 2019 war die Trennung der Marken fertig. OpenVAS repräsentiert nun den tatsächlichen Vulnerability Scanner, wie es auch ursprünglich war. Das Framework, indem sich OpenVAS befindet, ist das "Greenbone Vulnerability Management". Der OpenVAS Scanner, der mit GVM-10 veröffentlicht wurde, erhielt zahlreiche Performance-Optimierungen damit die ständig wachsende Zahl an Schwachstellen-Tests und Netzwerken besser verarbeitet werden kann. Mit GVM-11 wurden essentielle Adaptierungen im Framework vorgenommen. So wurde der openvassd in das command-line tool openvas tranformiert. Dadurch wurde ein neues stateless, XML basierte OSP (Open Scanner Protokoll) eingeführt. Die aktuelle Community Edition existiert in der Version 22.4, welche auch erstmalig den Notus Scanner einführte.

3.2 Architektur von OpenVAS

Wie in der Einleitung bereits erwähnt, besteht OpenVAS aus mehr als nur einem Scanner. Dennoch bildet dieser den Kern des Frameworks [7]. Die Hauptkomponenten werden von den folgenden 3 Kategorien gebildet.

- Vulnerability Scanner, die ausgewählte Zielsystem auf Schwachstellen testen
- Greenbone Vulnerability Management Daemon (gvmd)
- Greenbone Security Assistant (GSA) zusammen mit dem Greenbone Security Assistant Daemon (gsad)

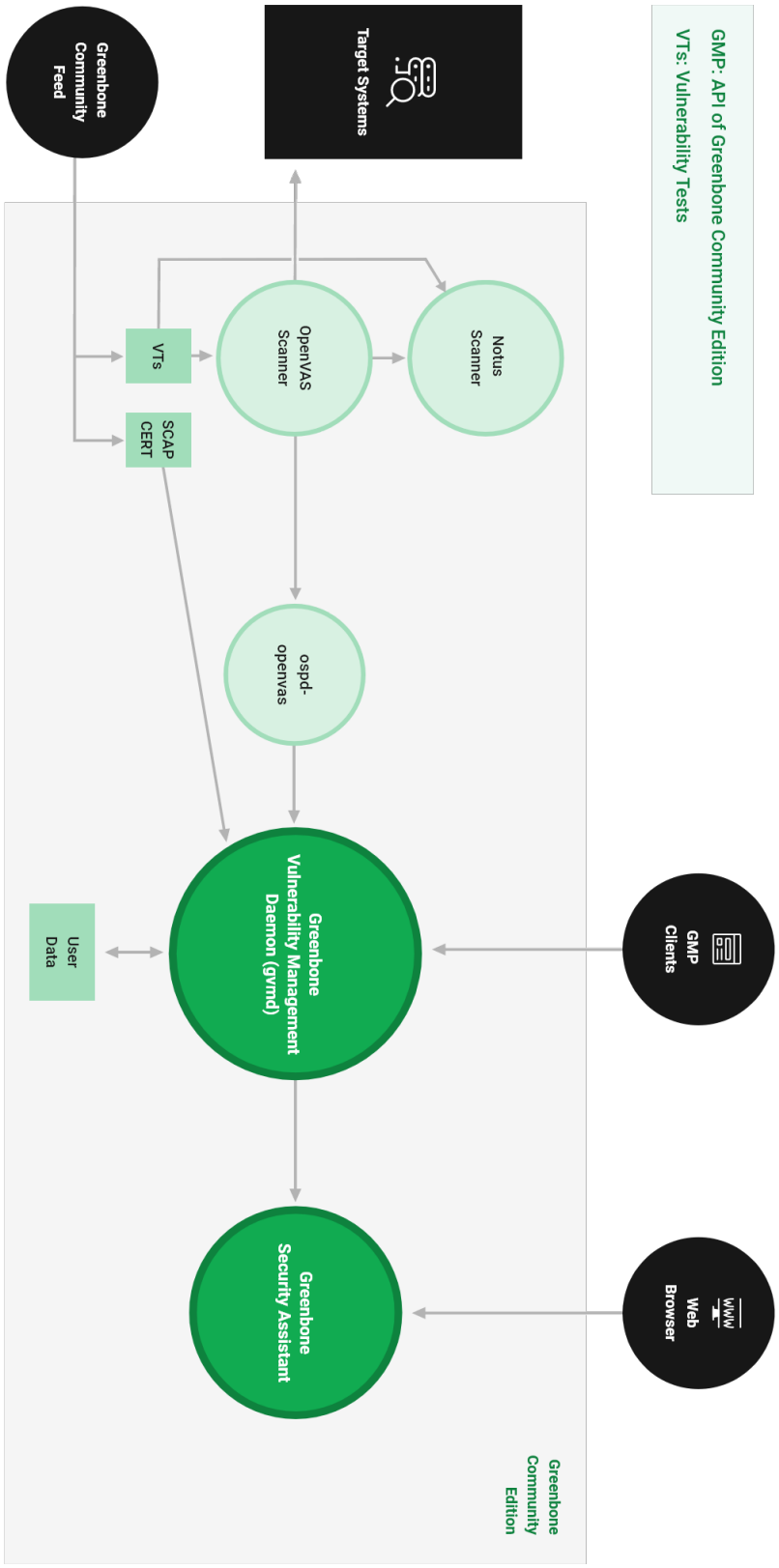

Abbildung 3.1: OpenVAS Architektur [3]

Die Abbildung 3.1 zeigt die verschiedenen Komponenten aus dem das Framework aktuell besteht. Dieses Framework unterlief mehreren Veränderungen und wurde über die Jahre stetig ergänzt. Im Folgenden werden die wichtigsten Komponenten der aktuellen Architektur des GVM-22 in der Community Edition näher erläutert:

Greenbone Vulnerability Management Daemon gvmd, ehemals OpenVAS Manager, bezeichnet den zentralen Dienst, der aus einem einfachen Scan-Ergebnis eine voll funktionsfähige Vulnerability Management Solution erzeugt [22]. Dabei wir über das Open Scanner Protocol (OSP) der OpenVAS Scanner gesteuert. Der Manager selbst kann über das XML basierte, zustandslose Greenbone Management Protocol (GMP) gesteuert werden. Konfigurationen und Scan-Ergebnisse werden in der zentralen SQL Datenbank verwahrt, die auch vom Manager kontrolliert wird. Weitere Merkmale des GVMd zeichnen sich durch User Management mit Rechteverteilung von Gruppen und Rolen sowie interne Laufzeitsysteme für geplante Aufgaben und andere Events aus.

Greenbone Security Assistant GSA bezeichnet einen Client, der als React Web-Service implementiert ist und dem User die Steuerung des Managers über ein User Interface ermöglicht. Neben der Webapplikation besteht der GSA auch aus einem HTTP Server (GSAD), der durch das GMP mit dem GVMd kommuniziert.

GVM-Tools können verwendet werden um das GMP durch eine Skriptsprache wie Python zu nutzen. Somit besteht als Gegenstück zur Weboberfläche des GSA die Möglichkeit den GVMd durch eine Skriptsprache zu kontrollieren.

VTs Aktuell existieren mehr als 100 000 VTs, Tendenz steigend. Der öffnetliche Feed wird als Greenbone Community Feed (GCF) bezeichnet und repräsentiert den kostenlosen Feed. Er ist Teil des Greenbone Enterprise Feed (GEF), welcher als Teil der kommerziellen Schiene des Greenbone Security Managers erworben werden kann. Beide Feeds werden täglich geupdated wobei der GEF bis zu 60% mehr verwertbare Exploits laut Greenbone AG identifizieren kann. Somit will Greenbone eine ausgewogene Balance zwischen kostenlosen und kostenpflichtigen Feed sicherstellen. Der lokale Feed kann entweder online durch den Befehl greenbone-feed-sync automatisch synchronisiert oder als Archivdatei heruntergeladen werden. Dabei ist streng empfohlen die Integrität der Dateien zu überprüfen. Dies geschieht mithilfe des GCF Transfer Integrity Zertifikats. Dies wird zum Beispiel durch den Synchronisationsbefehl automatisch überprüft.

Notus Scanner wurde wie schon erläutert erst in der Version 22.4 eingeführt und definiert einen Scanner, der ohne direkte User Interaktion während jeder regluären Überprüfung auch automatisch ausgeführt wird [2]. Dabei soll er wesentlich ressourcensparender sein, als auch eine bessere Performance liefern. Der Einsatz dieses Scanners ersetzt grundsätzlich die NASL-basierenden lokalen Sicherheitschecks(LSCs). Dies führt zu einer optimierten Effizienz, weil nun nicht mehr ein VT Skript für jedes LSC ausgeführt wird, sondern ein Vergleich der installierten Software auf einem Host gegen alle bekannten Schwachstellen einer Software. Im Vergleich dazu prüft der OpenVAS Scanner für jeden einzelnen Host jede einzelne NASL

LSC. Die Liste installierter Software wird beim Notus Scanner gleich geladen, wird aber direkt verglichen mit den bekannten Softwareschwachstellen. Dadurch muss der Notus Scanner die LSCs nicht ausführen, da die Information der bekannten Softwareschachstellen bereits in einer einzelnen Liste vorhanden ist und nicht für jedes NASL Skript individuell.

OpenVAS Scanner repräsentiert einen Scanner, der regelmäßige Updates erhält und als full-featured Scanner gilt. OpenVAS basiert auf einer Client-Server-Architektur. Nachdem der Server auf einem System ausgeführt wird, kann ein lokaler oder entfernter Client darauf zugreifen. Das simple Prinzip kann wie folgt erklärt werden: Der Client stellt eine Verbindung zum Server her und kann verschiedene Konfigurationen durchführen sowie Scans ausführen. Der Client erhält schließlich eine Übersicht der gefundenen Sicherheitslücken. [27] zeigt durch die folgende Abbildung die klassische OpenVAS - Netzwerkarchitektur:

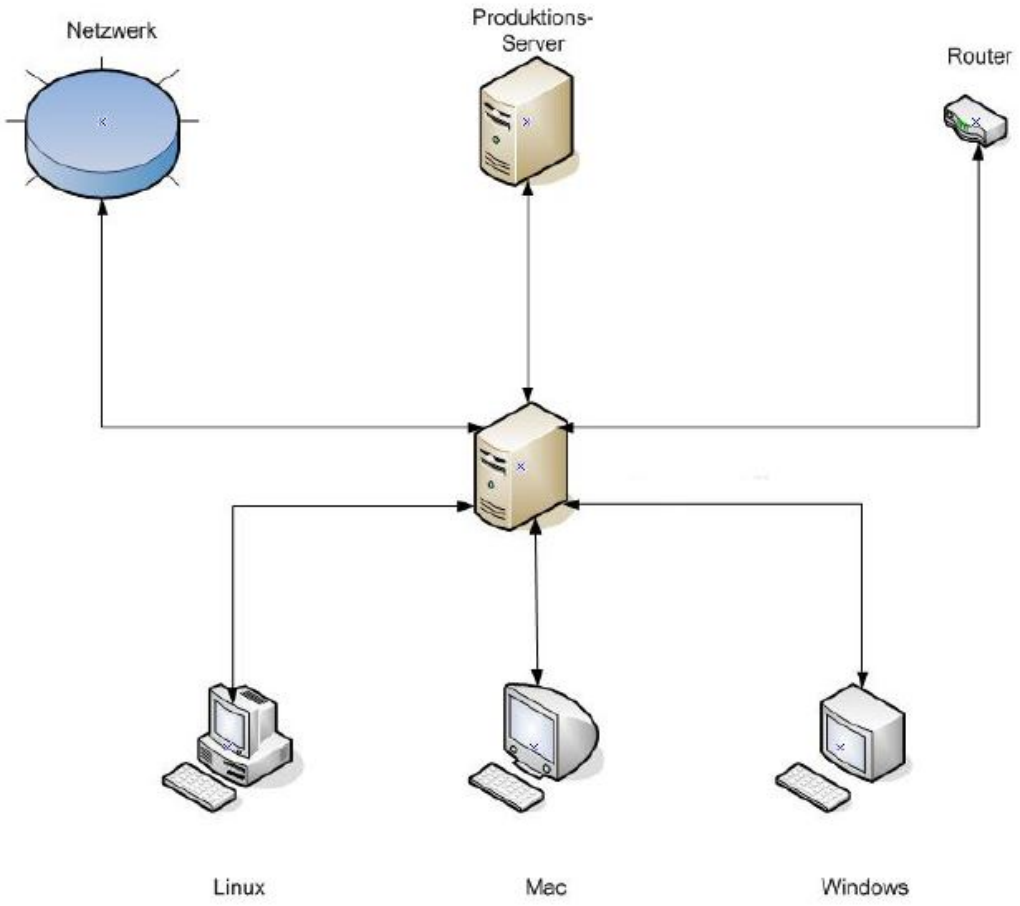

Abbildung 3.2: Typische OpenVAS Client-Server Architektur [27]

3.3 Funktionen

[7] liefert folgenden stichpunktartigen Überblick der Funktionalitäten von OpenVAS:

- Web-, QT- oder CLI basierte Bedienung möglich
- Security-Scanning von ganzen Netzwerken
- Durchführung lokaler, authentifizierter Sicherheitsprüfungen
- Einbindung verschiedener Sicherheits-Tools dank entsprechender Schnittstellen und Steuerprotokolle
- Zentrale Sammlung und Auswertung der Prüfergebnisse
- Hilfestellungen in Prüfberichten und über das Internet
- Zeitgesteuerte Prüfungen möglich
- Scan-Notizen ermöglichen die Kommentierung der Prüfergebnisse
- Umfangreiche Such-, Filter- und Sortierungsfunktionen in Prüfberichten
- False Positive-Umwidmung ermöglicht es, falsche Fehlererkennungen anders einzustufen
- Export von Reports in unterschiedliche Formate möglich (txt, pdf, xml, LaTex, html, cpe, itg, nbe) auch in topologische und geografische Visualisierung
- Prüfung von Web Anwendungen möglich
- Skalierbarer Master-Slave-Betrieb mit Scan-Sensoren möglich
- Nutzung individueller Richtlinienvorgaben
- IT-Grundschutz-Unterstützung
- Management von Scan-Aufgaben
- Unterstützung für Mehrbenutzer-Betrieb
- Vergleich von Berichten
- Export von Berichtsdaten in verschiedenen Formaten
- Inventarisierungs-Unterstützung
- Benutzeroberfläche -, GUI- und Web-basiert
- Integrierte Web Application Scans
- Integration von Security Content Automation Protocol (SCAP)-Standards
- Alarmierung bei Richtlinien-Verstoß oder Gefahr
- Manuelles oder automatisiertes Feed-Update von tausenden Network Vulnerability Tests (NVTs) über das Internet

3.4 OpenVAS - Setup

Um den Scanner sowohl im Modell zu berücksichtigen, als auch für die Implementierung des Prototypen zu verwenden, werde ich im Folgenden eine Vorgehensweise, verschiedene Herausforderungen und Probleme bei der Installation des OpenVAS Scanners anführen. Seit 2022 bietet Greenbone ein Dockerimage zur Installation der Greenbone Community Edition [13]. In einer frühen Phase dieser Arbeit verwendete ich die von Greenbone zur Verfügung gestellte ISO-Datei in einer virtuellen Maschine mit Ubuntu-Distribution [5]. Bei der Installation der notwendigen Pakete bis hin zum Start aller Komponenten, stellte ich mich zahlreichen Herausforderungen. Von Paketinkonsistenzen bis hin zur erfolgreichen Synchronisation der NVT-Feeds mussten einige Hürden überwunden werden. Mit dem Einsatz des Dockerimage wurde dies deutlich vereinfacht und schafft auf schnellerem Weg ein lauffähigeres System. Aber auch hier kristallisierten sich gewisse Probleme mit fehlerhaften NVT-Feed Syncs oder Datenbankmigration heraus. So musste im Zuge eines Updates während dieser Arbeit die Datenbank zu einer neuen Version migriert werden . Dadurch funktionierte die Synchronisation der NVT-Feeds nicht mehr und kein Scan konnte durchgeführt werden. Was folgte, war ein Durchsehen aller Logs aus den 17 Services, die im docker-compose.yml für den Start der Applikation enthalten waren.

3.5 Verwendung

Um die Funktionsweise und Bedienung von OpenVAS zu verstehen, gilt es sich zunächst einen Überblick über das System und die Möglichkeiten zu verschaffen. Abbildung 3.3 zeigt das Dashboard nach dem erfolgten Login im Browser.

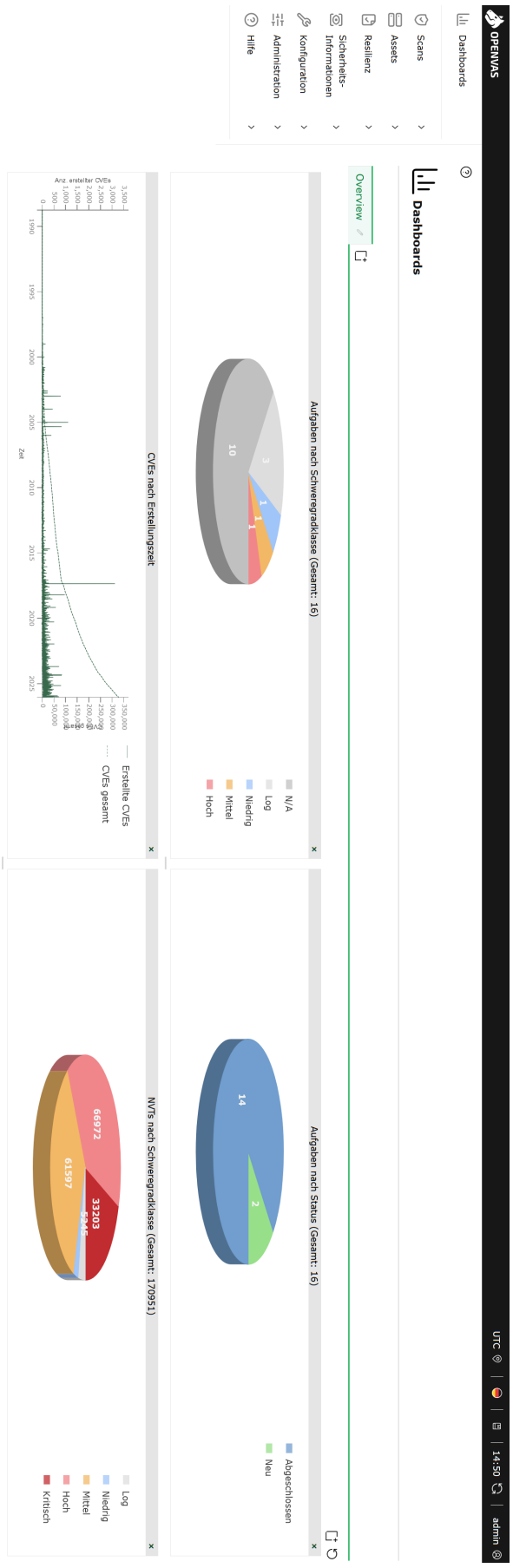

Abbildung 3.3: OpenVAS Dashboard

Um einen Scanvorgang starten zu können, bedarf es in erster Instanz die Ziele über die Konfiguration festzulegen. Dabei ist der zu durchsuchende Bereich zu definieren und gegebenenfalls Zugangsdaten einzugeben, um einen authentifizierten Scanvorgang starten zu können (Abbildung 3.4).

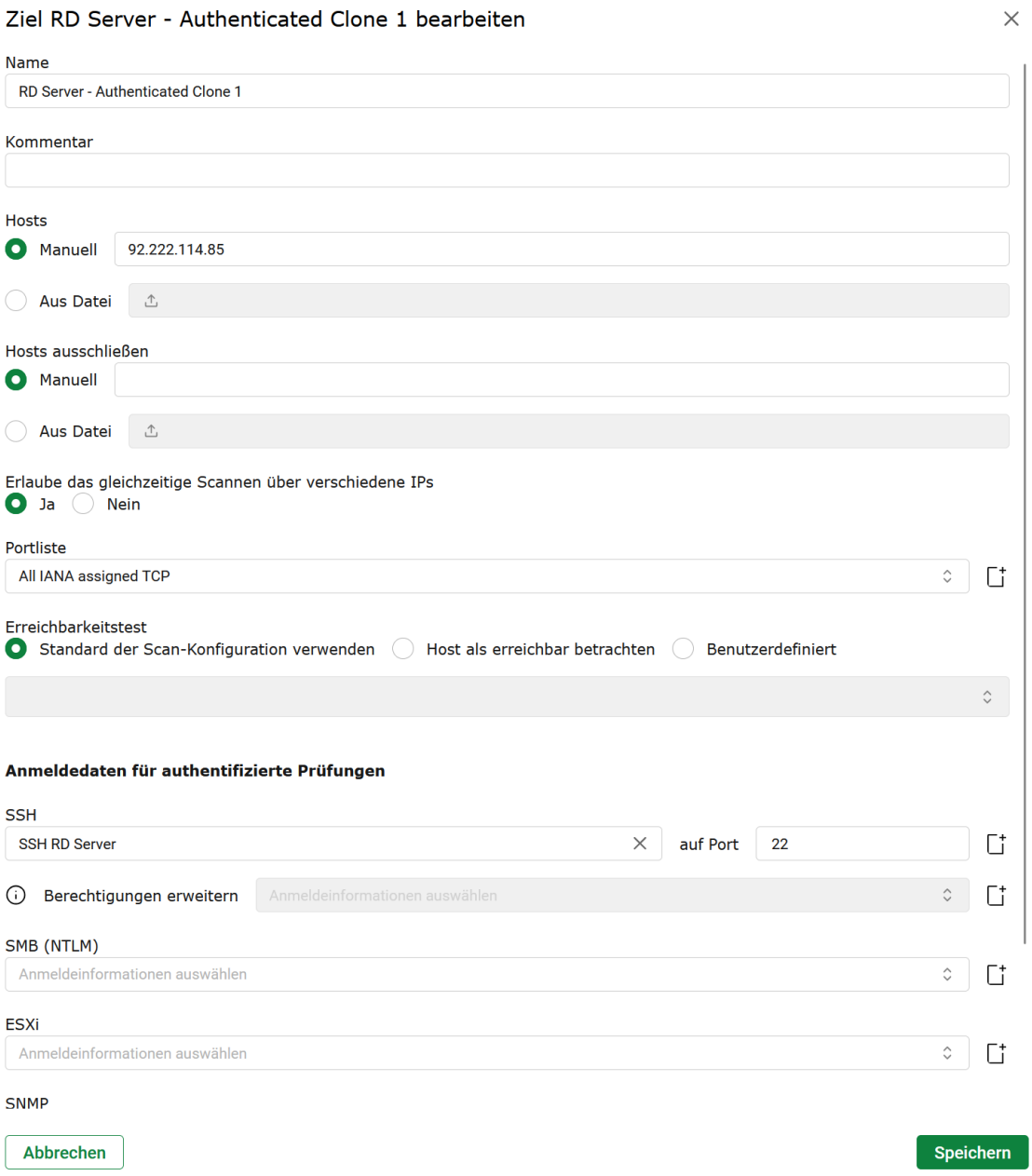

Abbildung 3.4: OpenVAS Scanzieldefinition

Anschließend kann über Scans->Tasks eine neue Scanaufgabe definiert werden (Abbildung 3.5). Neben dem Scanziel gilt es hier eine gewünschte Scankonfiguration auszuwählen. Diese

reicht von "Discovery" zu "Full and fast". Nach der Erstellung der Aufgabe wird diese in der Liste "Tasks" angezeigt (Abbildung 3.6). Dort existieren eine Reihe von Aktionen die für jede Aufgabe durchgeführt werden können.

- Starten, Stoppen, Löschen, Bearbeiten, Klonen und Exportieren

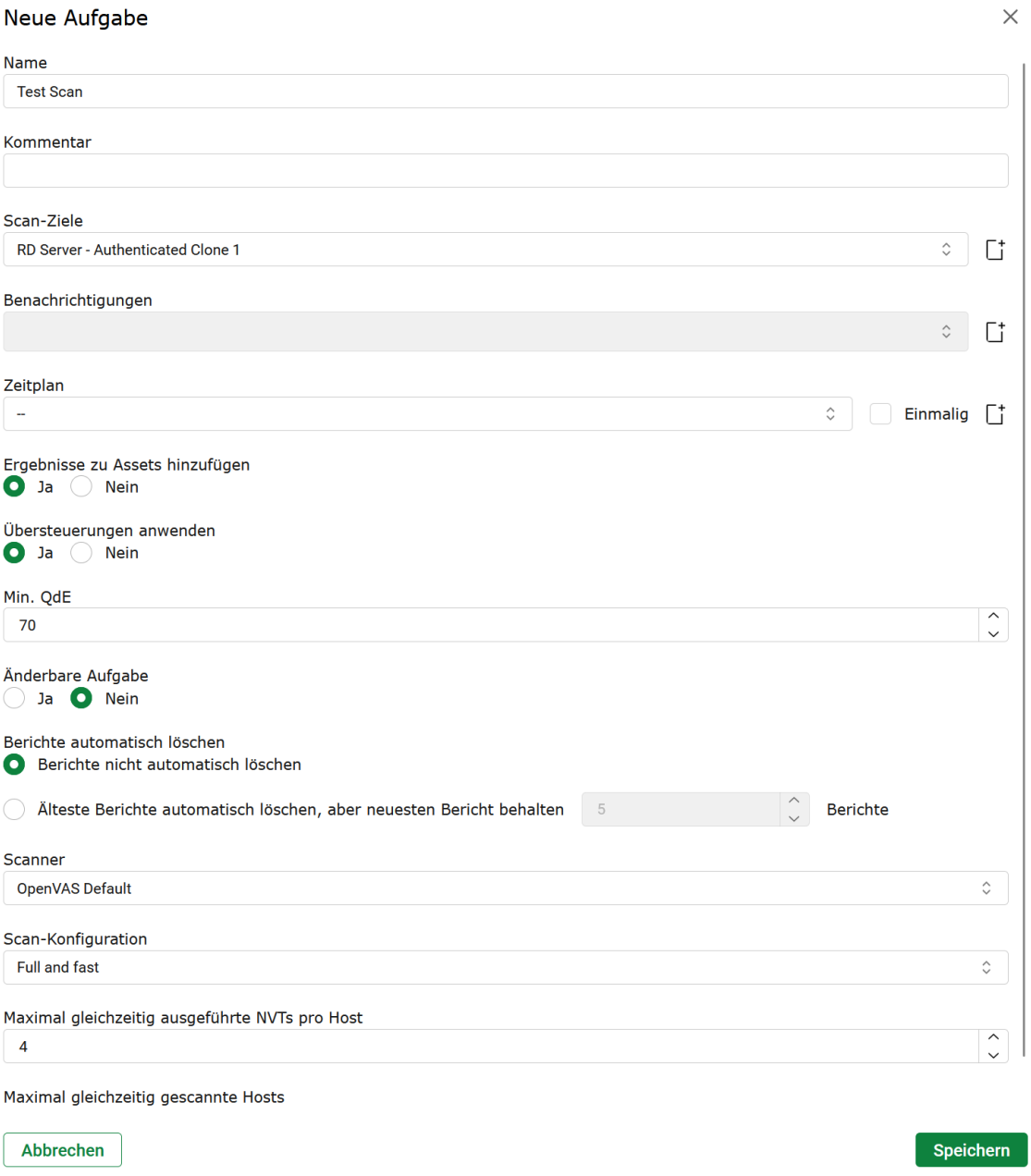

Abbildung 3.5: OpenVAS Scanaufgabe

Abbildung 3.6: OpenVAS Task Liste

Für jeden erfolgten Scan wird ein Report generiert, der die Ergebnisse des Scans zusammenfasst (Abbildung 3.7. Ein Klick auf eine gefundene Verwundbarkeit zeigt eine detaillierte Ansicht über diese (Abbildung 3.8).

Informationen | Ergebnisse (5 von 214) | Hosts (1 von 1) | Ports (2 von 6) | Anwendungen (33 von 33) | Betriebssysteme (1 von 1) | CVEs (3 von 3) | Geschlossene CVEs (0 von 0) | TLS-Zertifikate (3 von 3) | Fehlermeldungen (3 von 3) | Benutzer-Tags (0)

1 - 5 von 5

Schwachstelle	Schweregrad	QdE	Host IP	Name	Ort	EPSS Score	EPSS Perzentil	Erstellt
Ubuntu: Security Advisory (USN-6847-1)	8.8 (Hoch)	97 %	92.222.114.85		package	N/A	N/A	Sa., 18. Okt. 2025 09:47 Coordinated Universal Time
Ubuntu: Security Advisory (USN-5620-1)	6.5 (Mittel)	97 %	92.222.114.85		package	N/A	N/A	Sa., 18. Okt. 2025 09:47 Coordinated Universal Time
Ubuntu: Security Advisory (USN-6764-1)	5.0 (Mittel)	97 %	92.222.114.85		package	N/A	N/A	Sa., 18. Okt. 2025 09:47 Coordinated Universal Time
TCP Timestamps Information Disclosure	2.6 (Niedrig)	80 %	92.222.114.85		general/tcp	N/A	N/A	Sa., 18. Okt. 2025 10:01 Coordinated Universal Time
Weak MAC Algorithm(s) Supported (SSH)	2.6 (Niedrig)	80 %	92.222.114.85		22/tcp	N/A	N/A	Sa., 18. Okt. 2025 10:04 Coordinated Universal Time

Abbildung 3.7: OpenVAS Report

Abbildung 3.8: OpenVAS Report Detail

Ein Scanvorgang kann auch über externe Skripts durch das GMP-Protokoll konfiguriert und gestartet werden. Diese Funktion wird einen wichtigen Bestandteil des Prototypen in Kapitel 6 darstellen.

Authentifizierter und Nichtauthentifizierter Scanvorgang Um einen näheren Einblick in die Scantiefe von OpenVAS zu erlangen, ist es wichtig die Grenzen und Möglichkeiten eines Scans näher zu untersuchen. In unterschiedlichen Tests habe ich unauthentifizierte vs nichtauthentifizierte Scans durchgeführt. Dazu wurde ein Full and Fast Scan ausgeführt. Abbildung 3.9 zeigt dabei deutlich, dass ohne entsprechender Authentifzierung kein relevantes Testing möglich ist.

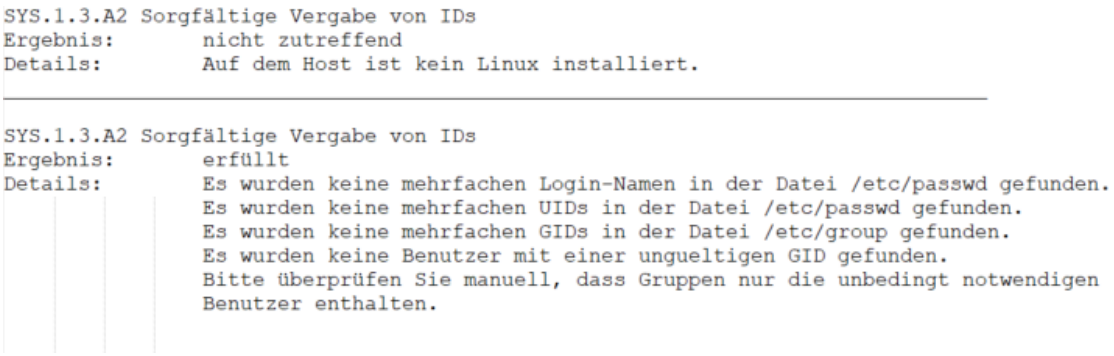

Abbildung 3.9: Authentifizierter VS Nichtauthentifizierter Scan

Damit ein authentifizierter Scan durchgeführt werden kann, müssen Zugangsdaten zum Beispiel in Form von SSH (Linux) oder SMB (Windows) vorliegen. Für linuxbasierte OS wie Ubuntu, genügt es den SSH Zugang zu aktivieren und die Zugangsdaten eines berechtigten Users in die Scankonfiguration einzutippen. Wesentlich komplizierter ist dieser Vorgang für

Windows Clients durchzuführen. Damit der Scanner einen detaillierteren Scan durchführen kann, muss unter anderem die Remoteregistrierung aktiviert und angepasst werden, sowie Anpassungen in der Registry durchgeführt werden. Für genaue Scanvorgänge ist es daher von Nöten, einen authentifizierten Scan durchführen zu können. Dies bezieht sich vor allem auch auf die Scans zur Feststellung eines geeigneten IT-Grundschutzes basierend auf das IT-Grundschutz-Kompendium. Dies wird in Kapitel 4.6 und Kapitel 5 näher erläutert.

Grundsätzlich erkennt man, dass bei Linux wesentlich mehr Informationen ausgelesen werden können als bei Windows. Dies lässt sich nicht zuletzt auch durch das quelloffene Linux erklären. Nichtauthentifizierte User dürfen unter Windows nicht auf das Registry zugreifen und können demnach keine Informationen über Patches oder Updates abrufen [24]. Um fehlerhaftes Verhalten am Zielsystem auszuschließen, wird die Scaneinstellung safe_checks = yes verwendet.

3.6 GMP - Python

Durch eine Python Bibliothek besteht die Möglichkeit den Scanner über das GMP Protokoll zu steueren [17]. Diese API unterstützt GMP und OSP mit drei verschiedenen Verbindungstypen, nämlich TLS, SSH und Unix Domain Socket. Der empfohlene Verbindungstyp für eine lokale Kommunikation ist der Unix Domain Socket. Als einfachstes Beispiel gilt die Abfrage nach der GMP Version. Angepasst an die installierte und konfigurierte Version, die auf dem Testserver installiert wurden, liefert das folgende Listing die GMP Version (Abbildung 3.10).

```
import sys

from gvm.connections import UnixSocketConnection
from gvm.errors import GvmError
from gvm.protocols.gmpv7 import Gmp
from gvm.transforms import EtreeCheckCommandTransform

path = '/var/run/openvasmd.sock'
connection = UnixSocketConnection(path=path)
transform = EtreeCheckCommandTransform()
gmp = Gmp(connection=connection, transform=transform)

try:
    with gmp:
        print(gmp.get_version())
except GvmError as e:
    print('An error occurred', e, file=sys.stderr)
```

Abbildung 3.10: GMP GetVersion()

Das Ausführen dieses Skripts liefert folgendes Ergebnis:

<get_version_response status="200" status_text="OK"><version>7.0
</version></get_version_response>

Dabei sieht man auch, dass das Ergebnis als XML String retouniert wird. Für die Implementierung des Prototypen wird XPATH eine wichtige Rolle einnehmen, um die benötigten Ergebnisse auszulesen. Das von Greenbone zur Verfügung gestellte Tech Doc Portal und die API ermöglicht es, sich über über die Parameterierung und Strukturierung der Requests und Responses zu informieren [23].

4 Bundesamt für Sicherheit in der Informationstechnik (BSI) und IT-Grundschutz

4.1 Allgemeine Informationen

Als Geschäftsbereich des Bundesministeriums des Innern wurde das Bundesamt für Sicherheit in der Informationstechnik (BSI) am 1. Januar 1991 gegründet [12]. Das BSI fungiert als unabhängige, neutrale Stelle für Angelegenheiten im Bereich IT-Sicherheit und ist als Behörde einzigartig im europäischen Raum. Den Hauptsitz besitzt das BSI in Bonn.

4.2 Aufbau / Zielsetzung

Das BSI ist in fünf Abteilungen unterteilt, die sich wiederum aus Fachbereichen zusammensetzen. Jeder Fachbereich besteht aus verschiedenen Referaten. Als nationale Cyber-Sicherheitsbehörde unterstützt das BSI durch Prävention, Detektion und Reaktion für Staat, Wirtschaft und Gesellschaft [12]. Somit hat sich das BSI den sicheren Einsatz von Informations- und Kommunikationstechnik als Ziel gesetzt. Sie wollen damit bewirken, dass Sicherheitsaspekte bereits bei der Entwicklung von IT-Systemen und -Anwendungen einbezogen werden. Sie richten sich mit Ihrem Angebot an Nutzer und Hersteller von Informationstechnik. Das BSI bietet seine Dienstleistungen in vier Kernbereichen an:

- **Information** über alle wichtigen Themen im Bereich der IT-Sicherheit
- **Beratung** in Bereich IT-Sicherheit und Unterstützung bei der Umsetzung geeigneter Maßnahmen
- **Entwicklung** von IT-Sicherheitsanwendungen und -produkte
- **Zertifizierung** und Prüfung von IT-Systemen hinsichtlich umgesetzter Sicherheitseigenschaften.

Neben unterschiedlichen BSI-Standards, stellt das BSI auch den IT-Grundschutz zur Verfügung. Im Rahmen dieser Arbeit gilt der Fokus auf das IT-Grundschutz-Kompendium, welches im folgenden Kapitel detailliert erläutert wird.

4.3 IT-Grundschutz-Kompendium

Das IT-Grundschutz-Kompendium wurde am 1.Februar 2018 in seiner ersten Version veröffentlicht und dient als Nachschlagewerk für Informationssicherheit [12]. Diese erste Version enthält 80 IT-Grundschutz-Bausteine, die in zukünftigen Editionen stetig erweitert und aktualisiert werden. Mit 1.Februar 2023 wurde die vorerst letzte Version des IT-Grundschutz-Kompendium veröffentlicht, die 111 Bausteine enthält. Diese Bausteine thematisieren unterschiedliche Bereiche der Informationssicherheit wie zum Beispiel Anwendungen (APP) oder Sicherheitsmanagement (ISMS) und sind in zehn Schichten aufgeteilt. Das IT-Grundschutz-Kompendium dient seit seiner Veröffentlichung als Prüfungsgrundlage für Zertifizierungen nach ISO 27001 auf Basis des IT-Grundschutz. Nach 6 Veröffentlichungen entschied sich das BSI für eine Weiterentwicklung des IT-Grundschutz++ [11]. Mögliche Fehler oder Ergänzungen werden seit 2023 in Form von Errata veröffentlicht. Ziel dieser neuen Entwicklung liegt in einer prozessorientierteren Anwendung des IT-Grundschutz.

4.3.1 Aufbau

Im Folgenden erläutere ich anhand von [12] die unterschiedlichen Bereiche des IT-Grundschutz-Kompendium.

Einstieg und Methodik
Dieser Bereich dient dazu sowohl die Struktur als auch den Einsatz des IT-Grundschutz-Kompendium zu verstehen. Des Weiteren dient dieser Bereich der Vorstellung eines Sicherheitskonzept nach IT-Grundschutz.

Hinweise zum Schichtenmodell und zur Modellierung
Für die korrekte Modellierung nach IT-Grundschutz bedarf es der Auswahl und Umsetzung der passenden Bausteine. Für die vereinfachte Auswahl existieren prozess- und systemorientierte Bausteine. Erstere werden für große Teile des Informationsverbund eingesetzt, System-Bausteine in der Regel auf einzelne Objekte oder Gruppen. Beide Bausteingruppen lassen sich wiederum in Teilschichten unterteilen. Die jeweiligen Hinweise zum Schichtmodell und zur Modellierung erklären, wann und wie ein Baustein eingesetzt werden soll und für welche Zielobjekte er benutzt werden kann. Bausteine werden nach Ihrer Umsetzungspriorität (vor- oder nachranging markiert.

Beschreibung der Rollen
Jeder Baustein enthält Rollen, die für dessen Umsetzung zuständig sind. Aufgrund der oft unterschiedlichen Bezeichnungen für Tätigkeiten in der IT-Branche gibt das IT-Grundschutz-Kompendium eine kurze Rollenbeschreibung.

Glossar und Begriffsdefinitionen
Das Glossar des IT-Grundschutz-Kompendium erläutert die wichtigsten Begriffe zur Informationssicherheit und IT-Grundschutz.

Elementare Gefährdungen
47 elementare Gefährdungen sind im IT-Grundschutz-Kompendium aufgelistet, dabei wurden die folgenden Ziele verfolgt. [12] beschreibt elementare Gefährdungen als Gefährdungen, die für die Verwendung bei der Risikoanalyse optimiert sind, soweit als möglich produkt- und technikneutral sind, Kompatibilität zu ähnlichen Katalogen und Standards aufweist und sich nahtlos in den IT-Grundschutz integriert.

Bausteine
Die Bausteine lassen sich, wie bereits erwähnt, in Prozess- und Systembausteine unterteilen (vgl. Abbildung 4.1), die einen exemplarischen Auszug der zugrunde liegenden Struktur zeigt [12, 8]. Diese Unterteilung bringt einige Vorteile mit sich. Durch eine sinnvolle Aufteilung der Sicherheitsaspekte, wird die Komplexität reduziert. Des weiteren werden Redundanzen verhindert, weil übergeordnete Aspekte und gemeinsame infrastrukturelle Problematiken getrennt von den IT-Systemen behandelt werden. Somit müssen diese Aspekte nur einmal abgearbeitet werden und nicht für jedes System einzeln. Das BSI erstellte die Schichten so, dass auch die Zuständigkeiten gebündelt sind. So sind zum Beispiel für die Schicht SYS die Zuständigkeiten für IT-Systeme verantwortlich und für die Schicht NET Netzwerkadministratoren. Ein weiterer Vorteil, der sich durch die Aufteilung in Schichten ergibt, ist, die resultierende einfachere Handhabung von Aktualisierungen und Erweiterungen von Schichten ohne, dass andere Schichten umfangreich davon betroffen sind.

- Prozess-Bausteine: Diese gelten grundsätzlich für alle Teile des Informationsverbunds.
  - ISMS (implementierte Anforderungen): Diese Schicht enthält den Baustein Sicherheitsmanagement, der in Folge als Grundlage für alle weiteren Tätigkeiten des Sicherheitsprozesses dient
  - ORP (Organisation und Personal): organisatorische und personelle Sicherheitsaspekte
  - CON (Konzepte und Vorgehensweisen): Bausteine, die sich konkret mit Konzepten und Vorgehensweisen beschäftigen, wie zum Beispiel der Baustein "Kryptokonzept"
  - OPS (Betrieb): Bausteine in dieser Schicht umfassen alle Sicherheitsaspekte der betrieblichen Art. Ein Beispiel dafür wäre der Baustein "Schutz vor Schadprogrammen". Dabei gelten diese Aspekte sowohl bei einer IT-Abteilung im Unternehmen als auch bei Outsourcing
  - DER (Detektion & Reaktion): Diese Schicht beinhaltet Bausteine, die sich um die Überprüfung der umgesetzten Maßnahmen, das Erkennen von Sicherheitsvorfällen sowie die ideale Reaktion dafür anzuwenden.

- System-Bausteine: Im Unterschied zu den Prozess-Bausteinen, werden System-Bausteine auf einzelne Zielobjekte oder Gruppen von Zielobjekten angewendet.
  - APP (Anwendungen): Die Absicherung von Anwendungen und Diensten in verschiedensten Bereichen wie Kommunikation, netzbasierte Dienste, Verzeichnisdiensten und viele mehr wird in dieser Schicht behandelt

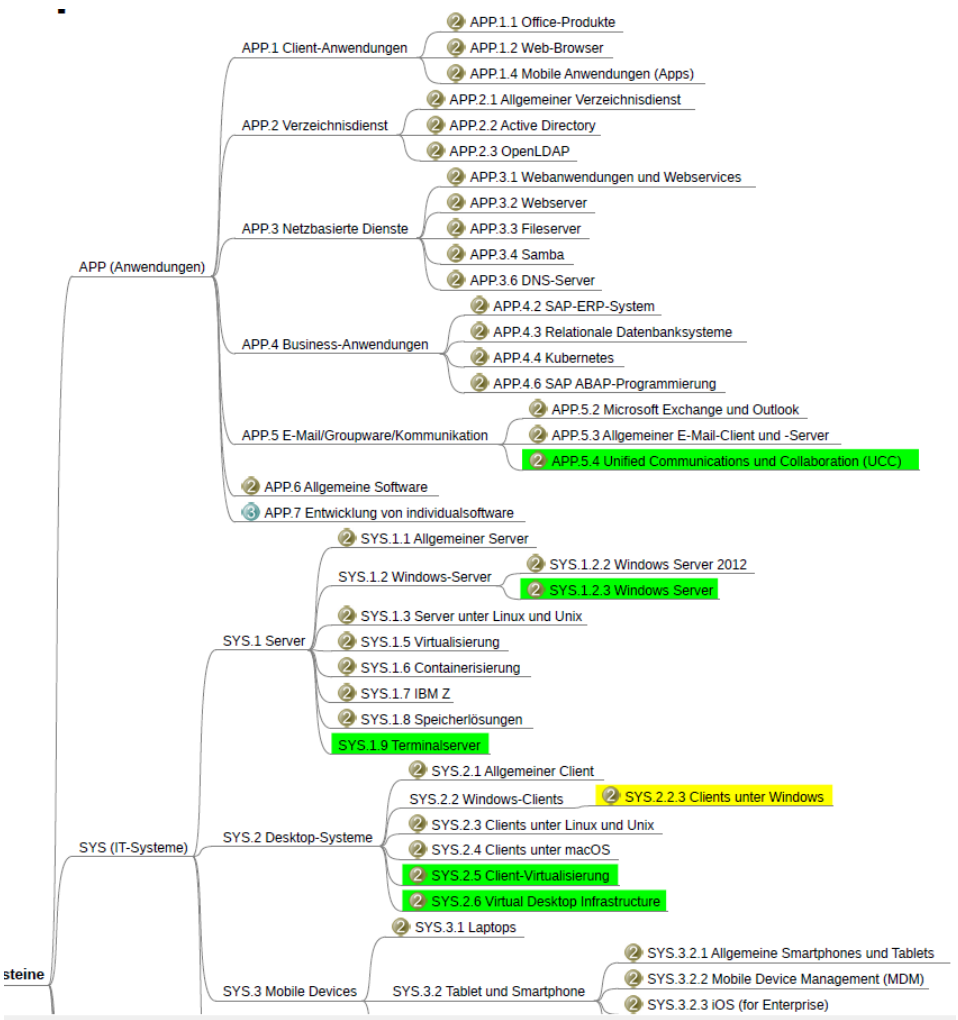

Abbildung 4.1: Auszug Struktur der Bausteine im IT-Grundschutz-Kompendium 2023 [12]

- SYS (IT-Systeme): Diese Schicht befasst sich mit einzelnen IT-Systemen, gegebenfalls auch Gruppen. Darunter fallen unter anderem Server, Mobile Devices oder Desktop-Systeme.
- IND (Industrielle IT): In dieser Schicht existieren Bausteine, die sich mit den Sicherheitsaspekten der industriellen IT beschäftigen wie zum Beispiel "Betriebs- und Steuerungstechnik"
- NET (Netze und Kommunikation): Hier befasst man sich primär mit der Kommunikation und Netzverbindungen anstelle von bestimmten IT-Systemen. Ein typischer Baustein dieser Schicht wäre zum Beispiel der Baustein "Firewall".
- INF (Infrastruktur): Diese Schicht beschäftigt sich mit der infrastrukturellen Sicherheit, die mit den baulich-technischen Gegebeneheiten in Verbindung stehen.

Jeder Baustein enthält eine kurze Beschreibung der untersuchten Komponenten, Vorgehensweisen und IT-Systeme, sowie für diese Komponenten spezifische Gefährdungen und deren konkreten Anforderungen, damit diese Komponente abgesichert werden kann. Das nachfolgende Kapitel erklärt den Bereich der Bausteine detaillierter.

4.3.2 Bausteine

Die Bausteine spielen eine zentrale Rolle des Nachschlagewerks, wobei der folgende Aufbau für alle Bausteine ident ist:

- Beschreibung des betrachteten Zielobjekts
- Formulierung der Zielsetzung, die den erreichten Sicherheitsgewinn nach Umsetzung des Bausteins erläutert
- Abgrenzung der Aspekte, die von diesem Bausteine nicht beachtet werden und Verweise auf Bausteine, die diese Aspekte aufgreifen
- Auflistung spezifischer Gefährdungen ohne Anspruch auf Vollständigkeit, die aufzeigen, welche Gefährdungen ohne Umsetzung der Gegenmaßnahmen entstehen können. Diese spezifischen Gefährdungen wurden aus den elementaren Gefährdungen abgeleitet
- Anschließend folgen die Anforderungen, gegliedert in drei Kategorien: Basis- und Standardanforderungen, sowie Anforderungen bei erhöhtem Schutzbedarf.
  - Basisanforderungen: Vorrangige Umsetzung; Können mit geringem Aufwand größtmöglichen Nutzen erzielen.
  - Standardanforderungen: Diese ermöglichen dem Unternehmen in Kombination mit den Basisanforderungen einen normalen Schutzbedarf.
  - Anforderungen bei erhöhtem Schutzbedarf: Neben den Basis- und Standardanforderungen werden dem Anwender Vorschläge für Anforderungen bei erhöhtem Schutzbedarf genannt.

- Beschreibung der Rollen
- Am Ende werden weiterführende Informationen und Verweise aufgelistet.
- Alle Bausteine werden in einem Anhang um eine Tabelle ergänzt, die die Anforderungen den betreffenden elementare Gefährdungen zuordnet.

Die Anforderungen sind nach folgendem Schema nummeriert, zB.: SYS.3.4.A2. Hier wird einerseits die Gruppe des Bausteins benannt, hier "SYS = System", anschließend die Nummern der Teilschichten und des Bausteins ("3.4") und schließlich die Anforderung ("A2"). Sofern Umsetzungshinweise zur aufgelisteten Anforderung existieren, finden sich diese unter der gleichen Nummer, jedoch mit vorangestelltem "M", zum Beispiel "SYS.3.4.M2".
Neben weiteren Rollen, die es geben kann, wird für jeden Baustein immer ein Haupt-Verantwortlicher benannt. Die Rollen werden in eckigen Klammern im Titel der Anforderung angeführt, zum Beispiel "CON.7.A9 Sicherer Umgang mit mobilen Datenträgern [Benutzende]." Dieses Beispiel einer Anforderung zeigt, dass der Benutzer selbst für diesen Baustein verantwortlich ist.

Modalverben werden eingesetzt um die Anforderungen eindeutig zu kennzeichnen.

- MUSS / DARF NUR: Anforderung muss erfüllt sein
- DARF NICHT / DARF KEIN: Auf keinem Fall darf die genannte Einstellung/Tätigkeit/Vorhaben umgesetzt werden
- SOLLTE: Anforderung sollte grundsätzlich umgesetzt werden, in Ausnahmefällen können aber Gründe existieren, die eine Umsetzung dieser Anforderung annuliert. Eine sorgfältige Abwägung muss vorliegen und durch Begründungen gestützt werden. Dies wird bei einem Audit überprüft.
- SOLLTE NICHT / SOLLTE KEIN: Entspricht dem Gegenteil der "SOLLTE" Anforderung, wobei hier Gründe bestehen können, die Anforderung dennoch umzusetzen.

Beispieleinsatz der Modalverben

- Basisanforderung: Bestehen aus den starken Modalverben "MUSS" und "DARF". Zum Beispiel kann eine Anforderung folgendermaßen formuliert werden: Der Administrator MUSS abc umsetzen und DARF NICHT jenes machen.
- Standardanforderung: Hier wird das Modalverb "SOLLTE" verwendet, das zu folgenden Anforderungen führt: Die Anforderung abc SOLLTE/SOLLTE NICHT erledigt werden.

Das Modalverb "SOLLTE" kann auch als Teilanforderung in Verbindung mit einer Basisanforderung stehen, um eine Ergänzung der MUSS-Anforderung darzustellen. Ein Beispiel dafür zeigt der Baustein OPS.1.2.2 Archivierung[8]:

> "Alle Zugriffe auf elektronische Archive MÜSSEN protokolliert werden. Dafür SOLLTEN Datum, Uhrzeit, Benutzer,Clientsystem und die ausgeführten Aktionen sowie Fehlermeldungen aufgezeichnet werden. [...]"

Diese Teilanforderung kann reziprok auch auf das Modalverb "MUSS" umgesetzt werden wie die Standard-Anforderung "ISMS.1.A10 Erstellung eines Sicherheitskonzepts" zeigt[8]:

> "Für den festgelegten Geltungsbereich (Informationsverbund) SOLLTE ein angemessenes Sicherheitskonzept als das zentrale Dokument im Sicherheitsprozess erstellt werden. [...] Im Sicherheitskonzept MÜSSEN aus den Sicherheitszielen der Institution, dem identifizierten Schutzbedarf und der Risikobewertung konkrete Sicherheitsmaßnahmen passend zum betrachteten Informationsverbund abgeleitet werden. [...]"

Kreuzreferenztabellen bezeichnen die Übersicht, die zeigt wie die elementare Gefährdungen des IT-Grundschutzes in Verbindung zu den Anforderungen des Bausteins stehen. Diese Kreuzreferenztabellen sind im Anhang des Bausteins ersichtlich und haben einen einheitlichen Aufbau [12].

- Kopfzeile: Elementare Gefährdungen des dazugehörigen Bausteins
- 1.Spalte: Die Nummern der Anforderungen
- Übrigen Spalten: Beschreibung der genauen Beziehung zwischen Anforderung des Bausteins und der elementaren Gefährdung
- Ein eingetragenes "X" bedeutet, dass die Anforderung gegen die Gefährdung wirksam ist, wobei diese Wirkung sowohl schadensvorbeugender als auch schadensmindernder Natur sein kann

Wichtig zu beachten ist, dass wenn eine spezifische Anforderung für keine Gefährdung relevant ist (kein "X"), dies keineswegs bedeutet, dass auf diese Anforderung verzichtet werden kann. Dies muss im Einzelfall überprüft werden, weil andere Gründe für diese Anforderung vorliegen könnte, die zum Beispiel in der Sicherheitskonzeption des Unternehmens festgelegt sind.

4.3.3 Umsetzungshinweise

Aktuell verfügen viele Bausteine über detaillierte Umsetzungshinweise. Neben einer Beschreibung über die Umsetzung der Anforderungen werden im Detail geeignete Sicherheitsmaßnahmen genannt. Diese Maßnahmen sollten jedoch an die Rahmenbedingungen der jeweiligen Institution angepasst werden. Die Umsetzungshinweise enthalten eine beispielhafte Umsetzung in Form eines Zyklus. Die Phasen des Zyklus werden im Folgenden nach [12] aufgelistet: "

- Planung und Konzeption
  - Definition des Einsatzzwecks
  - Festlegung von Einsatzszenarien
  - Abwägung des Risikopotentials
  - Dokumentation der Einsatzentscheidung
  - Erstellung des Sicherheitskonzepts

  - Festlegung von Richtlinien für den Einsatz
- Beschaffung (sofern erforderlich)
  - Festlegung der Anforderungen an zu beschaffende Produkte (nach Möglichkeit auf Basis der Einsatzszenarien der Planungsphase)
  - Auswahl der geeigneten Produkte
- Umsetzung
  - Konzeption und Durchführung des Testbetriebs
  - Installation und Konfiguration entsprechend Sicherheitsrichtlinie
  - Schulung und Sensibilisierung aller Betroffenen
- Betrieb
  - Sicherheitsmaßnahmen für den laufenden Betrieb (z. B. Protokollierung)
  - Kontinuierliche Pflege und Weiterentwicklung
  - Änderungsmanagement
  - Organisation und Durchführung von Wartungsarbeiten
  - Audit
- Aussonderung (sofern erforderlich)
  - Entzug von Berechtigungen
  - Entfernen von Datenbeständen und Referenzen auf diese Daten
  - Sichere Entsorgung von Datenträgern
- Notfallvorsorge
  - Konzeption und Organisation der Datensicherung
  - Nutzung von Redundanz zur Erhöhung der Verfügbarkeit
  - Umgang mit Sicherheitsvorfällen
  - Erstellen eines Notfallplans

"

Aufgrund laufender Weiterentwicklungen und Änderungen müssen manche Phasen immer wieder ausgeführt werden. Außerdem existieren nicht für jede Phase entsprechende Maßnahmen. Die Umsetzungshinweise sind nicht Teil des IT-Grundschutz-Kompendiums, sondern werden gesondert zu den Bausteinen als Hilfsmittel veröffentlicht.

4.3.4 Beispiel Baustein APP.3.1:Webanwendungen und Webservices

Im Folgenden wird exemplarisch der Aufbau des Bausteins APP.3.1. erläutert [6].

1. Beschreibung
    a) Einleitung
    b) Zielsetzung
    c) Abgrenzung
2. Gefährdungslage
    a) Unzureichende Protokollierung von sicherheitsrelevanten Ereignissen
    b) Offenlegung sicherheitsrelevanter Informationen bei Webanwendungen und Webservices
    c) Missbrauch einer Webanwendung durch automatisierte Nutzung
    d) Unzureichende Authentisierung
3. Anforderungen
    a) Basis-Anforderungen
        i. APP.3.1.A1 Authentisierung
        ii. APP.3.1.A4 Kontrolliertes Einbinden von Dateien und Inhalten
        iii. APP.3.1.A7 Schutz vor unerlaubter automatisierter Nutzung
        iv. APP.3.1.A14 Schutz vertraulicher Daten
    b) Standard-Anforderungen
        i. APP.3.1.A8 Systemarchitektur (S) [Beschaffungsstelle]
        ii. APP.3.1.A9 Beschaffung von Webanwendungen und Webservices
        iii. APP.3.1.A11 Sichere Anbindung von Hintergrundsystemen
        iv. APP.3.1.A12 Sichere Konfiguration
        v. APP.3.1.A21 Sichere HTTP-Konfiguration bei Webanwendungen
        vi. APP.3.1.A22 Penetrationstest und Revision
    c) Anforderungen bei erhöhtem Schutzbedarf
        i. APP.3.1.A20 Einsatz von Web Application Firewalls
4. Weiterführende Informationen
5. Anlage: Kreuzreferenztabelle zu elementaren Gefährdungen (Abbildung)

APP.3.1	Name	CIA	G 0.14	G 0.15	G 0.18	G 0.19	G 0.23	G 0.28	G 0.30	G 0.31	G 0.43	G 0.46
APP.3.1.A1	Authentisierung				X				X	X		
APP.3.1.A2	ENTFALLEN											
APP.3.1.A3	ENTFALLEN											
APP.3.1.A4	Kontrolliertes Einbinden von Dateien und Inhalten					X	X		X		X	
APP.3.1.A5	ENTFALLEN											
APP.3.1.A6	ENTFALLEN											
APP.3.1.A7	Schutz vor unerlaubter automatisierter Nutzung						X		X	X		
APP.3.1.A8	Systemarchitektur				X							
APP.3.1.A9	Beschaffung von Webanwendungen und Webservices				X			X				
APP.3.1.A10	ENTFALLEN											
APP.3.1.A11	Sichere Anbindung von Hintergrundsystemen			X		X	X				X	X
APP.3.1.A12	Sichere Konfiguration						X		X	X		
APP.3.1.A13	ENTFALLEN											
APP.3.1.A14	Schutz vertraulicher Daten					X						
APP.3.1.A15	ENTFALLEN											
APP.3.1.A16	ENTFALLEN											
APP.3.1.A17	ENTFALLEN											
APP.3.1.A18	ENTFALLEN											
APP.3.1.A19	ENTFALLEN											
APP.3.1.A20	Einsatz von Web Application Firewalls	CIA					X					
APP.3.1.A21	Sichere HTTP-Konfiguration bei Webanwendungen		X				X					
APP.3.1.A22	Penetrationstest und Revision				X			X				
APP.3.1.A23	ENTFALLEN											
APP.3.1.A24	ENTFALLEN											
APP.3.1.A25	ENTFALLEN											

Abbildung 4.2: Kreuztabelle APP.3.1 [12]

Zusätzlich gibt es Checklisten als Excel-Datei für jeden Baustein, der zur Verwendung im Betrieb angewendet werden kann. In früheren Versionen des Kompendiums existierten noch Umsetzungshinweise für den Baustein APP.3.1., die allerdings zu einem späteren Zeitpunkt wieder entfernt wurden. Diese wurden für diesen Baustein vermutlich aufgrund der technischen Schnelllebigkeit entfernt, um so den Usern veraltete Umsetzungshinweise zu ersparen. Grundsätzlich wurde von Version zu Version bei diesem Baustein einiges geändert. Maßnahmen, die zuvor für den erhöhten Schutz gegolten haben, wurden zum Beispiel in den Standardanforderungen übernommen. Auch eine Umbenennung des Bausteins von APP.3.1:Webanwendungen zu APP.3.1:Webanwendungen und Webservices erfolgte. Dies zeigt vor allem auch wie hoch der Pflegeaufwand ist beziehungsweise die schnell ändernde technologische Landschaft.

4.4 BSI und OpenVAS

OpenVAS wird vom BSI empfohlen und bietet zudem auch eine eigene Standardrichtlinie für das IT-Grundschutz-Kompendium an, die man automatisiert testen kann [18]. Lediglich die folgenden Bausteine können hierbei getestet werden:

- SYS.1.2.2 Windows Server 2012
- SYS.1.3 Server unter Linux und Unix
- SYS.2.2.2 Clients unter Windows 8.1
- SYS.2.2.3 Clients unter Windows 10
- SYS.2.3 Clients unter Linux und Unix

5 Adaptierter Vulnerability Management Life Cycle

Als Grundlage für die Konzeption eines geeigneten Ansatzes dient der Penetration-Testing-Life-Cycle, der in Kapitel 2.5 vorgestellt wurde. Dieser wurde im Rahmen der Literaturrecherche ausgewählt, da er einen etablierten und strukturierten Ablauf zur systematischen Identifikation und Analyse von Schwachstellen bietet. Insbesondere die klare Unterteilung in einzelne Phasen ermöglicht eine gezielte Anpassung und Erweiterung im Kontext dieser Arbeit. Alternative Ansätze im Bereich des Vulnerability Assessments wurden ebenfalls betrachtet, jedoch bietet der Penetration-Testing-Life-Cycle die notwendige Flexibilität, um sowohl automatisierte Scans als auch eine anschließende Weiterverarbeitung der Ergebnisse zu integrieren. Somit stellt er eine geeignete Grundlage für die Entwicklung eines erweiterten, auf die IT-Grundschutz-Kompendium abgestimmte Basis dar.

Ziel der vorliegenden Arbeit ist es, den bestehenden Life Cycle so zu erweitern, dass eine Verknüpfung zwischen automatisierten Scan-Ergebnissen und den Anforderungen des IT-Grundschutz-Kompendiums ermöglicht wird. Hierfür werden einzelne Phasen neu strukturiert und an die spezifischen Anforderungen dieses Projekts angepasst.

Eine wesentliche Anpassung betrifft die Phase *Exploitation*, die im klassischen Penetration Testing die aktive Ausnutzung identifizierter Schwachstellen beschreibt. Im Fokus der vorliegenden Arbeit steht jedoch nicht der praktische Gebrauch von Schwachstellen, sondern deren systematische Bewertung und Einordnung im Kontext des IT-Grundschutz-Kompendiums. Aus diesem Grund wird die Phase *Exploitation* durch die Phase *Mapping* ersetzt. Diese dient dazu, die im Rahmen der Schwachstellenanalyse identifizierten Befunde den jeweils relevanten Anforderungen des IT-Grundschutz-Kompendiums zuzuordnen. Auf diese Weise werden technische Scan-Ergebnisse in einen fachlich verwertbaren Zusammenhang überführt und für die weitere Bewertung nutzbar gemacht.

Der daraus resultierende adaptierte Life Cycle bildet die konzeptionelle Grundlage für das im weiteren Verlauf entwickelte Prozessmodell sowie für die spätere Implementierung des Prototyps. Die einzelnen Phasen des angepassten Life Cycles sind in Abbildung 5.1 dargestellt.

1. **Planning:** In dieser Phase werden die Rahmenbedingungen für den Scanprozess festgelegt. Insbesondere wird die Platzierung des Scanners innerhalb des Netzwerks definiert, da diese maßgeblich beeinflusst, welche Systeme erreichbar sind und welche Angriffsflächen sichtbar werden. Eine ungeeignete Positionierung kann zu unvollständigen oder verzerrten Ergebnissen führen und somit die Aussagekraft der späteren Bewertung er-

heblich beeinträchtigen.

2. **Reconnaissance:** Ziel dieser Phase ist die Identifikation aller im Netzwerk erreichbaren Systeme. Durch einen initialen Host-Scan wird eine Übersicht über potenzielle Zielsysteme geschaffen. Die Trennung dieser Phase von der detaillierten Analyse ermöglicht eine effizientere Durchführung, da unnötige Tiefenscans vermieden und die Belastung des Netzwerks reduziert werden.

3. **Exploration:** In der Explorationsphase erfolgt eine vertiefte Analyse der identifizierten Systeme. Dies kann durch erweiterte Scanmethoden oder die Nutzung von Zugangsdaten (z.B. authentifizierte Scans) erfolgen. Authentifizierte Scans ermöglichen dabei einen tieferen Einblick in Systemkonfigurationen und sind für die Überprüfung bestimmter Anforderungen zwingend erforderlich, da viele sicherheitsrelevante Informationen ohne entsprechende Berechtigungen nicht zugänglich sind.

4. **Vulnerability Assessment:** In dieser Phase werden die identifizierten Systeme systematisch auf Schwachstellen untersucht. Hierfür wird der OpenVAS-Scanner eingesetzt, dessen Konfiguration und Auswahl geeigneter Network Vulnerability Tests (NVTs) entscheidend für die Qualität und Aussagekraft der Ergebnisse ist. Standardkonfigurationen reichen hierbei nicht aus, da eine gezielte Anpassung an die Anforderungen des IT-Grundschutz-Kompendiums erforderlich ist.

5. **Mapping:** Die Mapping-Phase stellt eine zentrale Erweiterung des klassischen Life Cycles dar. Ziel ist es, die identifizierten Schwachstellen den entsprechenden Anforderungen des IT-Grundschutz-Kompendiums zuzuordnen. Dies ist notwendig, da Scan-Ergebnisse in der Regel technisch formuliert sind, während die Anforderungen des Kompendiums einen normativen Charakter besitzen. Das Mapping dient somit der Überführung technischer Befunde in einen fachlich interpretierbaren Kontext.

6. **Reporting and Recommendation:** In der abschließenden Phase werden die Ergebnisse strukturiert aufbereitet und bewertet. Rohbefunde aus Schwachstellenscans sind für Anwender oft nur eingeschränkt interpretierbar. Durch die Verdichtung der Ergebnisse und die Ableitung konkreter Handlungsempfehlungen wird ein praxisrelevanter Mehrwert geschaffen, der eine fundierte Bewertung des Sicherheitsniveaus ermöglicht.

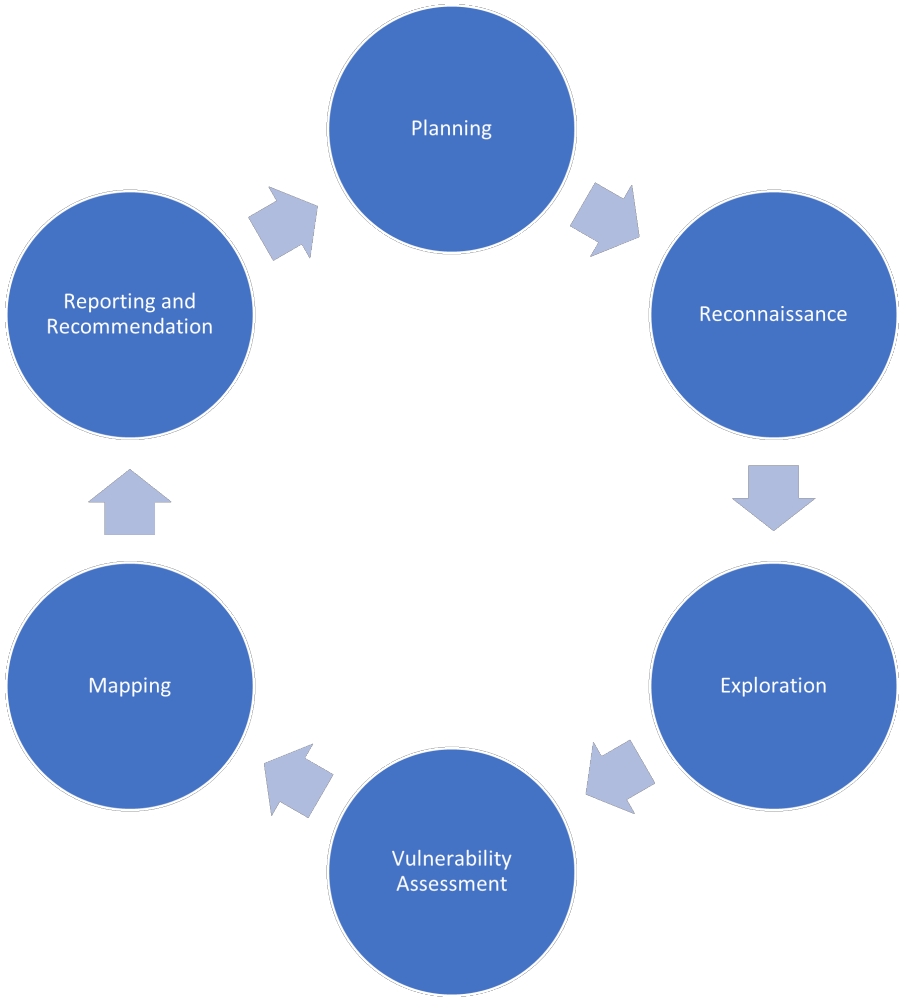

Abbildung 5.1: Modell: Adaptierter Testing Life-Cycle

5.1 Herausforderungen

Für die Entwicklung eines funktionsfähigen Prototypen ergeben sich mehrere zentrale Herausforderungen, die im Rahmen dieser Arbeit berücksichtigt werden müssen:

1. Das Mapping der Ergebnisse des Scanners mit dem IT-Grundschutz-Kompendium
2. Das Crawling des IT-Grundschutz-Kompendium, um die benötigte Information zu der gefundenen Verwundbarkeit zu finden
3. Die Konfiguration und Platzierung des OpenVAS Scanners im Netzwerk
4. Die geeignete Auswahl von NVTs.

Um ein ganzheitliches Modell erstellen zu können, bedarf es der Bewältigung dieser Herausforderungen. Im Folgenden werden diese nun detaillierter beschrieben:

5.1.1 Mapping

Ziel ist es, die von OpenVAS ermittelten Ergebnisse mit den Bausteinen des IT-Grundschutz-Kompendiums zu verknüpfen. Abbildung 5.2 repräsentiert ein Beispiel für ein Teilergebnis eines Scans. Basierend darauf ist ein Verfahren zu entwickeln, das eine eindeutige Zuordnung der Ergebnisse zu konkreten Anforderungen eines Bausteins ermöglicht. Auf dieser Grundlage kann anschließend bestimmt werden, ob ein Baustein als erfüllt oder nicht erfüllt zu bewerten ist.

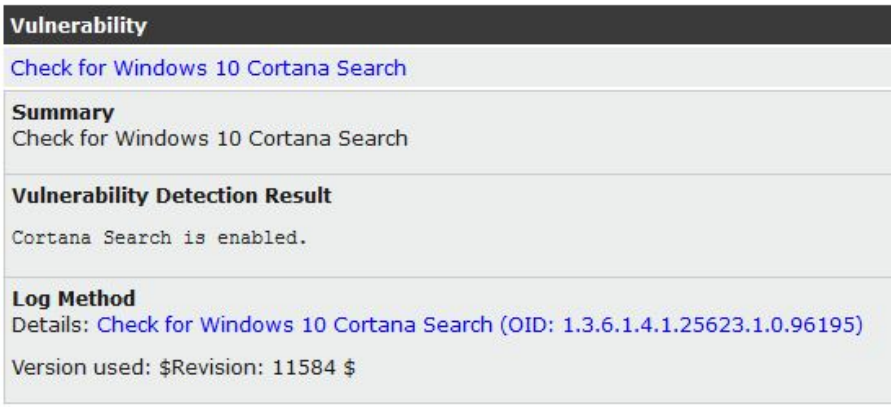

Abbildung 5.2: Ein Resultat eines authentifizierten Scans

5.1.2 Crawling

Um für die ermittelten Ergebnisse passende Informationen und Handlungsempfehlungen aus dem IT-Grundschutz-Kompendium abzuleiten, ist ein geeignetes Verfahren zur Datenextraktion zu entwickeln. Eine zentrale Herausforderung besteht dabei in den unterschiedlichen Formaten, in denen das Kompendium vorliegt. Dieses ist sowohl über die Webseite des BSI als auch als umfangreiches PDF-Dokument sowie in Form einer XML-Version verfügbar [12]. Daraus ergibt sich die Notwendigkeit, eine Lösung zu entwickeln, die eine einheitliche Extraktion relevanter Informationen aus den verschiedenen Formaten ermöglicht. Grundsätzlich eignet sich XML aufgrund seiner strukturierten Darstellung besonders gut für eine automatisierte Auswertung.

In der Praxis zeigt sich jedoch, dass die bereitgestellte XML-Version nur eingeschränkt nutzbar ist, da sie lediglich rudimentär strukturierte und semantisch wenig aussagekräftige Tags enthält. Offenbar handelt es sich hierbei um eine direkte Transformation der PDF-Struktur, bei der die ursprünglichen Bookmarks übernommen wurden. Dies führt dazu, dass keine generischen XPath-Ausdrücke verwendet werden können, um gezielt auf relevante Inhalte wie Anforderungen oder Beschreibungstexte zuzugreifen. Dadurch wird die Extraktion nicht zu einem rein technischen Parsing-Problem, sondern zu einem heuristischen Interpretationsproblem, bei dem relevante Inhalte anhand struktureller Muster identifiziert werden müssen.

Ein Auszug der XML-Struktur ist in Listing 5.1 dargestellt.

```
<section xml:id="scroll-bookmark-1595">
  <title>APP.3.1 Webanwendungen und Webservices</title>
  <section xml:id="scroll-bookmark-1596">
    <title>Beschreibung</title>
    <section xml:id="scroll-bookmark-1597">
      <title>Einleitung</title>
      <para>
        Webanwendungen bieten bestimmte Funktionen und dynamische
           ...
      </para>
    </section>
    <section xml:id="scroll-bookmark-1606">
        <title>Basis-Anforderungen</title>
    <para>Die folgenden Anforderungen SOLLTEN f r diesen
       Baustein vorrangig...</para>
        <section xml:id="scroll-bookmark-1607">
            <title>APP.3.1.A1 Authentisierung (B)</title>
                <para>Der IT-Betrieb MUSS Webanwendungen...</para
                   >
                <para>Der IT-Betrieb MUSS geeignete Grenzwerte
                   ...</para>
        </section>
    </section>
</section>
```

Listing 5.1: IT-Grundschutz XML-Struktur

5.1.3 OpenVAS

Die Integration des OpenVAS Scanners stellt eine weitere Herausforderung dar, insbesondere im Hinblick auf Installation, Konfiguration und flexible Einsetzbarkeit in unterschiedlichen Netzwerkumgebungen. Ziel ist es, eine Lösung zu schaffen, die eine einfache und reproduzierbare Durchführung von Scans ermöglicht. Die Grundeinstellungen des Scanners liefern dabei zum einen nur sehr wenige Möglichkeiten und zum anderen keine übersichtlichen Lösungen. Seit der Einführung des Docker-Images ist die Installation zwar erheblich erleichtert, dennoch zeigt Kapitel 3.4 Hürden, die das Image mit sich bringen kann. Zusätzlich sollten entsprechende Scankonfigurationen erstellt werden.

5.1.4 Auswahl NVT

Zur Überprüfung spezifischer Bausteine des IT-Grundschutz-Kompendiums ist die Auswahl geeigneter Tests sowie entsprechender Scankonfigurationen erforderlich. Eine vollständige Abdeckung aller Bausteine durch Network Vulnerability Tests (NVTs) ist jedoch nicht realisierbar.

Dies liegt zum einen an der großen Anzahl von Bausteinen (über 100 mit jeweils mehreren Anforderungen) und zum anderen daran, dass sich nicht alle Anforderungen automatisiert überprüfen lassen. Insbesondere ist es nicht möglich, einen NVT zu entwickeln, der allgemeingültig alle unnötigen Dienste und Funktionen eines Systems erkennt und bewertet. Darüber hinaus beziehen sich viele Anforderungen des IT-Grundschutz-Kompendiums auf organisatorische Maßnahmen, die grundsätzlich nicht automatisiert geprüft werden können.

5.2 Umsetzung

Aufgrund der großen Anzahl an Bausteinen sowie der damit verbundenen Anforderungen im IT-Grundschutz-Kompendium ist eine vollständige Betrachtung im Rahmen dieser Arbeit nicht realisierbar. Daher wird der Fokus gezielt auf ausgewählte Bausteine aus den Kategorien APP (Anwendungen), NET (Netzwerke) und SYS (IT-Systeme) gelegt.

Diese Auswahl wurde getroffen, da die genannten Kategorien einen Großteil der relevanten technischen Aspekte automatisierter Schwachstellenanalysen abdecken. Insbesondere lassen sich für diese Bausteine geeignete Prüfverfahren mithilfe von OpenVAS und entsprechenden Network Vulnerability Tests (NVTs) ableiten, wodurch eine praktische Umsetzung des entwickelten Ansatzes ermöglicht wird.

Im Anschluss an die Vorstellung der Lösungsansätze zur Bewältigung der identifizierten Herausforderungen werden die ausgewählten Bausteine sowie deren Anforderungen detailliert analysiert und in Bezug auf ihre technische Überprüfbarkeit bewertet.

5.2.1 Mapping der Ergebnisse

Das Mapping der Scan-Ergebnisse auf die Anforderungen des IT-Grundschutz-Kompendiums erfolgt in zwei Schritten.

Im ersten Schritt werden Ergebnisse aus bereits vorhandenen IT-Grundschutz-spezifischen Scankonfigurationen übernommen. Diese können ohne größeren Aufwand weiterverarbeitet werden, da sie bereits eine direkte Zuordnung zu entsprechenden Anforderungen enthalten. Die Ergebnisse lassen sich dabei im XML-Format exportieren und mithilfe von XPath analysieren und auswerten.

Im zweiten Schritt erfolgt die Auswertung von Ergebnissen anderer Scankonfigurationen, für die keine direkte Zuordnung zu Anforderungen des IT-Grundschutz-Kompendiums vorliegt. In diesem Fall ist eine manuelle oder heuristische Zuordnung der identifizierten Schwachstellen zu den entsprechenden Anforderungen erforderlich.

Ein Beispiel hierfür liefert die Scankonfiguration *Full and Fast*, bei der im Rahmen eines authentifizierten Scans eines Windows-Systems festgestellt wurde, dass der Sprachassistent Cortana aktiviert ist (Abbildung 5.2). Dieses Ergebnis kann der Anforderung *SYS.2.2.3.A14*

Einsatz des Sprachassistenten Cortana des Bausteins *SYS.2.2: Windows-Clients* zugeordnet werden. Ziel dieses Schrittes ist es, solche Zusammenhänge systematisch zu identifizieren und in eine konsistente Mapping-Struktur zu überführen. Die konkrete Umsetzung dieser Zuordnungslogik wird in Kapitel 5.3 detailliert beschrieben.

5.2.2 Crawling

Zur Lösung des Verknüpfungsproblems wird ein Ansatz basierend auf XPath in Kombination mit regulären Ausdrücken (RegEx) eingesetzt. Ziel ist es, relevante Informationen aus dem IT-Grundschutz-Kompendium automatisiert zu extrahieren und für das Mapping nutzbar zu machen.

Die Extraktion erfolgt heuristisch anhand der Bezeichnungen von Bausteinen und Anforderungen. Hierbei werden strukturelle Muster innerhalb der XML-Daten genutzt, um gezielt auf die gewünschten Inhalte zuzugreifen. Ein wesentlicher Vorteil dieses Ansatzes liegt darin, dass die IT-Grundschutz-Kompendien der vergangenen Jahre weitgehend einheitlich aufgebaut sind.

Dadurch kann auf eine aufwändige Datenmigration oder einen vollständigen Import in eine Datenbank verzichtet werden. Stattdessen ermöglicht das direkte Auslesen der jeweils aktuellen XML-Datei eine effizientere und wartungsärmere Verarbeitung der Daten. Die konkrete Implementierung sowie die detaillierte Vorgehensweise werden im Kapitel 6 Prototyp näher erläutert.

5.2.3 Integration von OpenVAS

Die Integration des OpenVAS Scanners erfordert eine geeignete Hardwareumgebung, da sowohl die Installation als auch die Konfiguration mit einem nicht unerheblichen Ressourcenbedarf verbunden sind.

Für kompakte Einsatzszenarien bietet sich grundsätzlich der Einsatz ressourcenschonender Hardware wie eines Raspberry Pi an. Allerdings zeigt sich, dass insbesondere bei größeren Netzwerken oder umfangreicheren Scankonfigurationen schnell Leistungsgrenzen erreicht werden können.

Im Rahmen dieser Arbeit wird daher eine leistungsfähigere Hardwarelösung eingesetzt. Konkret erfolgt die Implementierung des Prototyps auf einer OpenVAS-Instanz, in einem Docker-Container betrieben wird.

5.2.4 Auswahl von NVTs und Definition der Scankonfiguration

Zur Bewältigung der beschriebenen Herausforderungen wird eine angepasste Scankonfiguration entwickelt, die sowohl bestehende Prüfroutinen für das IT-Grundschutz-Kompendium

als auch zusätzliche, gezielt ausgewählte Network Vulnerability Tests (NVTs) integriert.

Ziel dieser Konfiguration ist es, eine möglichst umfassende Abdeckung relevanter Anforderungen zu erreichen, ohne dabei die Effizienz und Praktikabilität der Scans zu beeinträchtigen. Während vorhandene IT-Grundschutz-spezifische Prüfungen direkt übernommen werden können, ist für weitere Anforderungen eine manuelle Auswahl geeigneter NVTs erforderlich.

Anhand des zuvor dargestellten Beispiels der aktivierten Cortana-Funktion lässt sich dieser Ansatz verdeutlichen. Um diese Schwachstelle zu identifizieren, wird der NVT *"Check for Windows 10 Cortana Search"* (OID: 1.3.6.1.4.1.25623.1.0.96195) in die Scankonfiguration integriert.

Die Auswahl der NVTs erfolgt somit gezielt auf Basis ihrer Relevanz für spezifische Anforderungen des IT-Grundschutz-Kompendiums und stellt einen zentralen Bestandteil des entwickelten Ansatzes dar.

5.3 Ergebnisse

Dieses Kapitel beschreibt die im Rahmen der Arbeit betrachteten Bausteine des IT-Grundschutz-Kompendiums sowie die zugehörigen Anforderungen. Darüber hinaus wird erläutert, inwieweit diese Anforderungen durch das entwickelte Modell überprüft werden können.

Es ist hervorzuheben, dass der entwickelte Ansatz keine vollständige automatisierte Konformitätsprüfung gegenüber dem IT-Grundschutz-Kompendium darstellt. Vielmehr ermöglicht er eine technische Vorbewertung ausgewählter Anforderungen anhand maschinell erfassbarer Indikatoren. Organisatorische, prozessuale und dokumentationsbezogene Anforderungen können durch diesen Ansatz nur eingeschränkt oder gar nicht überprüft werden. Der Prototyp ist daher als Unterstützungssystem zur Identifikation und Priorisierung von Schwachstellen zu verstehen, nicht als Ersatz einer vollständigen IT-Grundschutz-Prüfung.

Zur Bewertung des Erfüllungsgrads werden drei Status definiert: *„erfüllt“*, *„nicht erfüllt“* und *„kann nicht automatisiert geprüft werden“*. Diese Klassifikation ermöglicht eine differenzierte Einordnung der Ergebnisse im Hinblick auf ihre technische Überprüfbarkeit.

Kapitel 4.6 gibt einen Überblick über die verfügbaren Prüfroutinen sowie die damit verbundenen Einschränkungen. Zur Erweiterung und Verbesserung dieser Prüfroutinen werden zusätzliche Network Vulnerability Tests (NVTs) aus der Community-Datenbank herangezogen und den entsprechenden Anforderungen zugeordnet. Dieser Schritt stellt einen wesentlichen Bestandteil des entwickelten Ansatzes dar.

Insgesamt stehen über 100.000 NVTs zur Verfügung, die unterschiedliche sicherheitsrelevante Aspekte von IT-Systemen abdecken. Die Auswahl der in dieser Arbeit betrachteten

Bausteine erfolgt unter Berücksichtigung ihrer technischen Überprüfbarkeit sowie der Rahmenbedingungen der eingesetzten Testumgebung.

Der Fokus liegt dabei auf den folgenden Bausteinen des IT-Grundschutz-Kompendiums:

- APP: Anwendungen
  - APP.3 Netzbasierte Dienste
    - APP.3.1 Webanwendungen und Webservices
    - APP 3.2 Webserver
- SYS: IT-Systeme
  - SYS.1 Server
    - SYS.1.3 Server unter Linux und Unix
- NET: Netze und Kommunikation
  - NET.1 Netze
    - NET3.1 Router und Switches

Von den ausgewählten Bausteinen werden die jeweiligen Anforderungen extrahiert und mit geeigneten Network Vulnerability Tests (NVTs) verknüpft.

OpenVAS stellt hierfür im Menübereich *SecInfo -> NVTs* eine Übersicht der im Community Feed verfügbaren Tests bereit. Diese können anhand verschiedener Kriterien gefiltert werden, beispielsweise über eine eindeutige Identifikationsnummer (OID) oder die Zugehörigkeit zu einer bestimmten Testfamilie.

Im Folgenden werden die betrachteten Bausteine sowie die Zuordnung der NVTs zu den jeweiligen Anforderungen des IT-Grundschutz-Kompendiums detailliert erläutert. Dabei ist zu berücksichtigen, dass einzelne NVTs für mehrere Anforderungen herangezogen werden können, da die gewonnenen Informationen häufig für unterschiedliche Anforderungen relevant sind.

5.3.1 SYS.1.1 Allgemeiner Server

Das IT-Grundschutz-Kompendium bezeichnet einen allgemeinen Server als ein IT-System mit einem beliebigen Betriebssystem, das Benutzern und anderen IT-Systemen gewisse Dienste bereitstellt [10]. Eine Vielzahl von Servern bildet den Kern vieler Unternehmen, sei es im eigenen Serverraum oder ausgelagert zu einem externen Hoster. Dabei existieren zahlreiche Gefährdungen, die es bestmöglich zu verhindern gilt. Die Palette reicht dabei von Softwarefehlern und Datenverlust bis hin zu Denial-of-Service-Angriffen, Überlastung von Servern oder der Bereitstellung unnötiger oder nicht benötigter Betriebssystemkomponenten, die letztendlich wieder eine Tür für Angriffe darstellen. Die Überprüfung dieses Bausteins wird über den Baustein SYS.1.3. im nächsten Kapitel betrachtet.

5.3.2 SYS.1.3: Server unter Linux und Unix

Dieser Baustein fokussiert sich auf das häufig eingesetzte Betriebssystem Linux oder Unix. Hieß dieser Baustein in den vorherigen Versionen noch "Server unter Unix" wurde er 2020 umbenannt, weil die Anforderungen vorrangig für Linux-Server adressiert sind. Zahlreiche kritische Geschäftsprozesse sowie Applikationen werden meist auf Linux-Servern abgebildet, deshalb kommt diesem Baustein auch eine besondere Bedeutung zu [10]. Die auf dem Server laufenden Systeme werden dabei als eigener Baustein behandelt (zum Beispiel APP.3.2 Webserver) und werden deshalb nicht betrachtet. Die Anforderungen dieses Bausteins beziehen sich auf den Schutz der zu bearbeitenden Informationen auf Betriebssystemebene. Als besondere Bedrohungen gelten das Ausspähen von System- und Benutzerinformationen, die Ausnutzbarkeit der Skriptumgebung, das dynamische Laden von gemeinsam genutzten Bibliotheken sowie die Nutzung von Software aus Drittquellen. Als authentifizierter User lassen sich hierbei über SSH mit Rootzugriff schon einige Bausteine automatisiert testen. Dabei können folgende Anforderungen durch die IT-Grundschutz Einstellung von OpenVAS ermittelt werden [10]:

- SYS.1.3.A2 Sorgfältige Vergabe von IDs
- SYS.1.3.A3 Kein automatisches Einbinden von Wechsellaufwerken
- SYS.1.3.A4 Schutz vor Ausnutzung von Schwachstellen in Anwendungen
- SYS.1.3.A5 Sichere Installation von Software-Paketen

Um die automatisierte Ermittlung anderer Anforderungen zu unterstützen, wurden ausgewählte NVTs ermittelt, die weitere Anforderungen überprüfen können.

- SYS.1.3.A6 Verwaltung von Benutzenden und Gruppen
- SYS.1.3.A8 Verschlüsselter Zugriff über Secure Shell
- SYS.1.3.A14 Verhinderung des Ausspähens von Informationen über das System und über Benutzende
- SYS.1.3.A17 Zusätzlicher Schutz des Kernels

SYS.1.3.A2 Sorgfältige Vergabe von IDs gibt die Benutzer und Gruppenkonfigurationen für Server vor. So darf zum Beispiel jeder Login-Name, Gruppe oder Benutzer nur einmal vorkommen. Zusätzlich gibt es einige MUSS Voraussetzungen, die erfüllt sein müssen - wie beispielsweise die Zuweisung von Benutzenden zu einer Gruppe oder die Definition jeder GID in der /etc/group. Außerdem muss bei vernetzten System auf eine Konsistenz bei Benutzer und Gruppen geachtet werden.

SYS.1.3.A3 Kein automatisches Einbinden von Wechsellaufwerken besagt die Unterbindung automatisch eingebundener Wechseldatenträger wie zum Beispiel USB-Sticks oder CDs.

SYS.1.3.A4 Schutz vor Ausnutzung von Schwachstellen in Anwendungen schreibt die zwingende Aktivierung von ASLR und DEP/NX im Kernel vor. Zusätzlich dürfen gewisse Sicherheitsfunktionen des Kernels nicht aktiv sein (zum Beispiel Heap- und Stackschutz).

SYS.1.3.A5 Sichere Installation von Software-Paketen Software, die direkt aus dem Quellcode kompiliert und installiert werden soll, darf nur von einem unprivilegierten Benutzerkonto entpackt, kompiliert und installiert werden. Zudem darf es nicht unkontrolliert in das Wurzelverzeichnis installiert werden. Außerdem sollten in diesem Fall die gewählten Parameter ausreichend dokumentiert werden, damit die Software nachvollziehbar und reproduzierbar kompiliert werden kann.

SYS.1.3.A8 Verschlüsselter Zugriff über Secure Shell fordert eine verschlüsselte und authentisierte Verbindung zwischen zwei IT-Systemen. Andere Dienst, abseits SSH, sollten vollständig deaktiviert werden. Zusätzlich sollten Zertifikate gegenüber Passwörter verwendet werden. Die Anforderung ist grundsätzlich technisch überprüfbar, da sich Konfigurationen und unterstützte kryptographische Verfahren automatisiert analysieren lassen. Eine vollständige Bewertung ist jedoch nur eingeschränkt möglich, da organisatorische Aspekte wie Schlüsselmanagement oder Benutzerverwaltung nicht vollständig durch einen Scanner erfasst werden können.

Zur technischen Überprüfung wurden folgende NVTs herangezogen:

- OID 1.3.6.1.4.1.25623.1.0.105610 - Weak MAC Algorithm(s) Supported (SSH): Zeigt, ob der Server schwache Client to Server Algorithmen unterstützt.
- OID 1.3.6.1.4.1.25623.1.0.105565 - SSH Protocol Algorithms Supported: Zeigt unterschiedliche Algorithmen, die vom SSH Service unterstützt werden
- OID 1.3.6.1.4.1.25623.1.0.117839 - Diffie-Hellman Ephemeral Key Exchange DoS Vulnerability: Zeigt eine SSH Service Akzeptanz der Diffie-Hellman ephemeral (DHE) Key Exchange (KEX), die anfällig für Denial-of-Service Angriffen ist

SYS.1.3.A14 Verhinderung des Ausspähens von Informationen über das System und über Benutzende Bei der Benutzung sollten möglichst wenige und nur notwendige Informationen sowie Log- und Konfigdateien nach außen zum Anwender gelangen. Weiters sollte darauf geachtet werden, dass es bei Befehlen zu keiner Weitergabe von sensiblen Inhalten kommt. Die eingesetzten NVTs identifizieren vorhandene Dienste, Softwarekomponenten und Systeminformationen. Diese können potenziell zur Preisgabe sensibler Informationen beitragen. Allerdings stellen die Ergebnisse keinen direkten Nachweis dar, dass tatsächlich sicherheitsrelevante Informationen nach außen offengelegt werden. Vielmehr handelt es sich um Indikatoren, die auf mögliche Informationslecks hinweisen und im Kontext interpretiert werden müssen.

- OID 1.3.6.1.4.1.25623.1.0.117232 - Apache HTTP Server Detection Consolidation: Zeigt, ob ein Apache Webserver verwendet wird
- OID 1.3.6.1.4.1.25623.1.0.105937 - OS Detection Consolidation and Reporting: Gibt Auskunft über das gefundene OS des gescannten Systems

- OID 1.3.6.1.4.1.25623.1.0.103592 - PHP Detection (Linux/Unix SSH Login): Zeigt, ob PHP Bibliotheken installiert sind

SYS.1.3.A17 Zusätzlicher Schutz des Kernels Um den Gebrauch von Sicherheitslücken bzw. möglichen Angriffspunkten zu verhindern, ist es empfohlen, Kernels oder andere Präventionsmaßnahmen (z.B. Dateiabsicherung, Speicherschutz) zu treffen.

- OID 1.3.6.1.4.1.25623.1.0.150584 - SYS.1.3.A4: Die Wiederverwendung dieses NVT ist sachlich vertretbar, da die zugrunde liegende Anforderung SYS.1.3.A4 eng mit der Absicherung des Kernels verknüpft ist. Ein negativer Befund kann daher als technischer Indikator für eine unzureichende Kernel-Härtung interpretiert werden.

5.3.3 SYS.2.2.3 Clients unter Windows

Seit Windows 8.1 setzt Microsoft neben lokalen Anwendungen auch auf cloudbasierte Dienstleistungen. Dies öffnet im Vergleich zur Vorgängerversion eine weitere potentiell gefährliche Tür für Angreifer. Die Hauptbedrohungen definiert das BSI wie folgt: Schadprogramme unter Windows 10, Integrierte Cloud-Funktionen, Beeinträchtigung von Software-Funktionen durch Kompatibilitätsprobleme sowie Telemetrie-Funktionen von Windows 10. Als Cloud-Funktion gilt zum Beispiel der von Microsoft entwickelte sprachgestützte Suchassistent "Cortana". Durch die Telemetrie-Funktionen von Windows 10 werden Diagnosedaten fortlaufend an Microsoft gesendet. Dies schließt beispielsweise den Zugriff auf die Registry des Clients mit ein [9]. Obwohl die IT-Grundschutz-Kompendium Scankonfiguration von OpenVAS einige Anforderungen anbietet, können diese aufgrund der Zugriffsbeschränkungen von Windows Systemen nicht getestet werden. Aus diesem Grund müssen die Anforderungen durch NVT selbst gewählt werden. Dadurch können folgende Anforderungen ermittelt werden:

- SYS.2.2.3.A4 Telemetrie und Datenschutzeinstellungen unter Windows

5.3.4 APP.3.1 Webanwendungen und Webservices

Grundlegend werden Webanwendungen durch das IT-Grundschutz-Kompendium [10] definiert als Anwendungen, die durch die Verwendung des HTTP(S) Protokolls Daten für andere Anwendungen bereitstellen. Das klassische Beispiel stellen hier Websites dar, die von einem Webserver Inhalte an diverse Browser senden. Eine Webanwendung beinhaltet dabei meist unterschiedliche Hintergrundsysteme (z.B. Datenbanksysteme) sowie einen Webserver, der den Inhalt des Servers an das angefragte System zurücksendet (z.B. Apache / Nginx). Das BSI hat sich hierbei auf REST(=Representational State Transfer) bezogene Applikationen und dessen Betrieb in diesem Baustein festgelegt. Webanwendungen sind in deren Einsatzgebiet sehr unterschiedlich einsetzbar beziehungsweise werden für die unterschiedlichsten Szenarien verwendet. Eine allgemeingültige Richtlinie oder umzusetzende Maßnahmen lässt sich hier schwierig definieren, weil aufgrund der zu verarbeitenden Datenmengen und Art von Daten, unterschiedliche sicherheitsrelevante Umsetzungen zu treffen sind. So sind sensible Daten einer medizinischen Einrichtung wesentlich kritischer zu betrachten und zu schützen

als beispielsweise öffentliche Termine eines Konzerts. Daher wird hier auf die (zumeist) allgemeingültigen Anforderungen Stellung bezogen. Als Ergebnis gilt die folgende Auflistung, die zeigt, wie die Anforderungen überprüft werden können. Dabei wurden die NVT nach eigenem Ermessen ausgewählt. Im Folgenden wird jeder der oben genannten Bausteine auf verfügbare NVTs überprüft und entsprechend zugeordnet.

- Durch Grundschutz-Konfiguration
  - keine passende Einstellung vorhanden
- Durch Auswahl geeigneter NVTs
  - APP.3.1.A1 Authentisierung
  - APP.3.1.A9 Beschaffung von Webanwendungen und Webservices
  - APP.3.1.A12 Sichere Konfiguration
  - APP.3.1.A21 Sichere HTTP-Konfiguration bei Webanwendungen

APP.3.1.A1 Authentisierung wird vom IT-Grundschutz-Kompendium beschrieben als eine Sicherheitsmaßnahme, die geschützte Ressourcen vor der Öffentlichkeit oder nicht autorisierten Diensten/Personen schützt. Aufgrund der Tatsache, dass nicht alle Webanwendungen geschützte Ressourcen besitzen, wird beim Scan im Kapitel 6 eine Auswahlmöglichkeit zum Scan dieser Anforderung vorhanden sein. Überprüft wird die Anforderung durch folgende Maßnahmen:

- OID 1.3.6.1.4.1.25623.1.0.108440 Cleartext Transmission of Sensitive Information via HTTP: Zeigt an, ob der Host sensitive Informationen (Zugangsdaten) unverschlüsselt via HTTP überträgt
- OID 1.3.6.1.4.1.25623.1.0.153974 Allowed HTTP Methods Enumeration: Zeigt an welche HTTP Methoden unterstützt werden.

APP.3.1.A9 Beschaffung von Webanwendungen und Webservices Diese Anforderung beschreibt organisatorische und technische Aspekte bei der Auswahl und Einführung von Webanwendungen. Eine vollständige automatisierte Überprüfung ist nicht möglich, da die Anforderung primär organisatorische Prozesse betrifft. Technische Schwachstellen können jedoch als Indikatoren für Defizite im Beschaffungs- oder Wartungsprozess interpretiert werden.

Zur Bewertung wurden folgende NVTs herangezogen:

- OID 1.3.6.1.4.1.25623.1.0.119125 WordPress <= 6.8.2 Multiple Vulnerabilities (Sep 2025) - Linux: Wichtig wenn die Beschaffung bzw. Installation von Wordpress im Fokus liegt, sollte darauf geachtet werden, dass die aktuellste Version installiert wird
- OID 1.3.6.1.4.1.25623.1.0.124887 WordPress Elementor Website Builder Plugin < 3.30.3 Path Traversal Vulnerability: Auch hier gilt es die aktuellste Version zu installieren, weil bei älteren Versionen Sicherheitslücken bekannt sind.

APP.3.1.A12 Sichere Konfiguration Bei der Einrichtung von Webanwendungen und -services sollte darauf geachtet werden, dass der Zugriff nur über sichere Kommunikationspfade möglich oder zumindest weitestgehend eingeschränkt ist. Dies wird gewährleistet, indem man einerseits keine für die Sicherheit relevanten Inhalte in Fehlermeldungen ausgibt, nicht benötigte HTTP-Methoden sperrt und andererseits die Zugriffsversuche beschränkt. Die eingesetzten NVTs liefern primär Informationen über die eingesetzten Technologien und Systemumgebung. Diese stellen keinen direkten Nachweis für eine sichere Konfiguration dar, sondern dienen lediglich als technische Indikatoren, die Rückschlüsse auf mögliche Fehlkonfigurationen zulassen.

- OID 1.3.6.1.4.1.25623.1.0.117232 - Apache HTTP Server Detection Consolidation: Zeigt, ob ein Apache Webserver verwendet wird
- OID 1.3.6.1.4.1.25623.1.0.105937 - OS Detection Consolidation and Reporting: Gibt Auskunft über das gefundene OS des gescannten Systems

APP.3.1.A21 Sichere HTTP-Konfiguration bei Webanwendungen Diese Anforderung setzt die Verwendung geeigneter HTTP-Header voraus. Dazu zählen

- Content-Security-Policy
- Content-Type
- X-Content-Type-Options
- Cache-Control

Zusätzlich sollten Cookies die Attribute secure, SameSite und httponly gesetzt haben.

Diese Anforderung ist technisch gut überprüfbar, da HTTP-Header und Cookie-Einstellungen direkt analysiert werden können. Die ausgewählten NVTs ermöglichen eine gezielte Prüfung sicherheitsrelevanter Header sowie von Cookie-Attributen. Die Ergebnisse liefern in diesem Fall vergleichsweise direkte Hinweise auf Fehlkonfigurationen, insbesondere bei fehlenden Sicherheitsheadern oder unsicheren Cookie-Einstellungen. Ausgewählte NVTs:

- OID 1.3.6.1.4.1.25623.1.0.105879 - SSL/TLS: HTTP Strict Transport Security (HSTS) Missing: Zeigt, ob der Server HTTP Strict Transport Security (HSTS) erzwingt
- OID 1.3.6.1.4.1.25623.1.0.105925 - Missing 'HttpOnly' Cookie Attribute (HTTP): Gibt Auskunft darüber, ob der Server das HTTPOnly Cookie gesetzt hat oder nicht.
- OID 1.3.6.1.4.1.25623.1.0.112081 - HTTP Security Headers Detection: Überprüft alle bekannten Security Header, die der Webserver liefert.

5.3.5 APP.3.2 Webserver

Ein Webserver stellt das Hauptstück eines jeden Webauftritts dar, weil er sozusagen als Schnittstelle zwischen Serveranwendungen und Clients fungiert. Die Inhalte werden dabei über das Hypertext Transfer Protocol (HTTP) oder der verschlüsselten Form HTTPS übermittelt. Da Webserver im Internet frei zugänglich sind und auch innerbetrieblich oftmals zur Daten- und Informationsweitergabe verwendet werden, ist die Absicherung dieser von immenser Bedeutung. Das BSI grenzt Webbrowser oder Netzwerkarchitekturen sowie die allgemeinen Server in diesem Baustein aus und verweist dabei auf die entsprechenden Bausteine. Aufgrund der hohen Gefährdungslage von Reputationsverlust über Manipulation bis hin zum Verlust vertraulicher Daten oder gar dem Verstoß gegen Gesetze oder Regelungen ist dieser Baustein in heutigen IT-Umgebungen ein sehr wichtiges Kriterium, weil er oftmals das Tor zur Außenwelt widerspiegelt. Folgend nun die Auflistung mit den gewählten NVTs zu diesem System:

- Durch Grundschutz-Konfiguration
  - keine passende Einstellung vorhanden
- Durch Auswahl geeigneter NVTs
  - APP.3.2.A1 Sichere Konfiguration eines Webservers
  - APP.3.2.A11 Verschlüsselung über TLS
  - APP.3.2.A12 Geeigneter Umgang mit Fehlern und Fehlermeldungen
  - APP.3.2.A13 Zugriffskontrolle für Webcrawler
  - APP.3.2.A14 Integritätsprüfungen und Schutz vor Schadsoftware

APP.3.2.A1 Sichere Konfiguration eines Webservers Nach der Installation eines Servers muss eine sichere Konfiguration vorgenommen werden. Hierbei sollten alle nicht benötigten Berechtigungen und Funktionen entzogen werden und die Konfiguration am besten in einem gesonderten, geschützten Bereich bzw. sogar, falls nicht anders möglich, auf einem eigenen virtuellen Server.

Ausgewählte NVTs:

- OID 1.3.6.1.4.1.25623.1.0.153974 - Allowed HTTP Methods Enumeration: Zeigt an welche HTTP Methoden unterstützt werden.
- OID 1.3.6.1.4.1.25623.1.0.10757 - Webmin / Usermin Detection (HTTP): Zeigt, ob der Dienst Webmin/Virtualmin auf dem Webserver läuft. Beim Einsatz von Webmin / Usermin sollte der Webmin Dienst nicht von außen erreichbar sein, weil hier zentrale Konfigurationen am Server vorgenommen werden können.

APP.3.2.A11 Verschlüsselung über TLS Der Webserver muss eine sichere Verschlüsselung über TLS (HTTPS) aufweisen, um Zugriffe von nicht vertrauenswürdigen Netzwerken abzuwehren. Wenn Anfragen mittels HTTPS-Verbindung einlangen, müssen auch Antworten über HTTPS versendet werden. Sollte dies aufgrund Schnittstellenkonflikten oder Verbindungsproblemen nicht möglich sein, können in Ausnahmefällen veraltete Versionen verwendet werden,

solange sich dies auf ein Minimum beschränkt. Die Anforderung ist grundsätzlich technisch gut überprüfbar, da sich sowohl unterstützte Protokollversionen als auch kryptographische Verfahren automatisiert analysieren lassen. Die ausgewählten NVTs decken verschiedene sicherheitsrelevante Aspekte ab, darunter veraltete TLS-Versionen, unsichere Cipher Suites sowie bekannte Angriffsszenarien.

Kein einzelner Test liefert jedoch einen vollständigen Nachweis für eine sichere TLS-Konfiguration. Erst die Kombination mehrerer Prüfschritte ermöglicht eine belastbare Bewertung. Die Ergebnisse sind daher als technische Indikatoren zu verstehen, die gemeinsam interpretiert werden müssen.

Ausgewählte NVTs zur technischen Überprüfung:

- OID 1.3.6.1.4.1.25623.1.0.105879 - SSL/TLS: HTTP Strict Transport Security (HSTS) Missing: Zeigt, ob der Server HTTP Strict Transport Security (HSTS) erzwingt
- OID 1.3.6.1.4.1.25623.1.0.117274 - SSL/TLS: Deprecated TLSv1.0 and TLSv1.1 Protocol Detection: Zeigt veraltete TLS Versionen an, die deaktiviert werden sollten
- OID 1.3.6.1.4.1.25623.1.0.117840 - Diffie-Hellman Ephemeral Key Exchange DoS Vulnerability (SSL/TLS, D(HE)ater): Gefahr einer DoS Attacke durch Unterstützung des Diffie-Hellman ephemeral (DHE) Algorithmus
- OID 1.3.6.1.4.1.25623.1.0.108147 - SSL/TLS: Report 'Anonymous' Cipher Suites: Webserver sollte so konfiguriert sein, dass er keine anonymen Verschlüsselungsmechanismen zulässt
- OID 1.3.6.1.4.1.25623.1.0.117414 - SSL/TLS: BREACH attack against HTTP compression: Zeigt die Verwundbarkeit einer SSL/TLS Verbindung zu BREACH (Browser Reconnaissance & Exfiltration via Adaptive Compression of Hypertext) an.
- OID 1.3.6.1.4.1.25623.1.0.103140 - SSL/TLS: Certificate - Self-Signed Certificate Detection: Zeigt ein selbtsigniertes Zertifikat an.

APP.3.2.A12 Geeigneter Umgang mit Fehlern und Fehlermeldungen Bei Fehlermeldungen sollten ausschließlich allgemeine Informationen enthalten sein, welche einen Fehler anzeigen. Es sollten keine Inhalte zu Produktname, Version oder System- bzw. Konfigurationsdaten ersichtlich sein.

Folgende ausgewählte NVTs liefern Hinweise auf eingesetzte Technologie und Systeminformationen, die potenziell in Fehlermeldungen enthalten sein können:

- OID 1.3.6.1.4.1.25623.1.0.117232 - Apache HTTP Server Detection Consolidation: Zeigt, ob ein Apache Webserver verwendet wird.
- OID 1.3.6.1.4.1.25623.1.0.105937 - OS Detection Consolidation and Reporting: Gibt Auskunft über das gefundene OS des gescannten Systems

APP.3.2.A13 Zugriffskontrolle für Webcrawler Jegliche Inhalte sollten so geschützt sein, dass sie Webcrawler abschirmen, welche sich nicht an den Robots-Exclusion-Standard halten.

Der ausgewählte NVT gibt dabei Rückschluss über das Vorhandensein einer robots.txt Datei:

- OID 1.3.6.1.4.1.25623.1.0.10302 robot.txt / robots.txt exists on the Web Server (HTTP): Die Robots.txt sollte so konfiguriert werden, damit keine schadhaften Bots oder Crawler relevante Ressourcen auslesen können.

APP.3.2.A14 Integritätsprüfungen und Schutz vor Schadsoftware IT-Betriebe sollten regelmäßige Kontrollen an Webservern vornehmen, um Informationen darüber zu erhalten, ob Konfigurationen und veröffentlichte Dateien weiterhin korrekt dargestellt werden.

- OID 1.3.6.1.4.1.25623.1.0.154915 - Apache HTTP Server 2.4.17 < 2.4.64 DoS Vulnerability - Linux: Zeigt die Verwundbarkeit des Apache Servers für DoS Angriffe an
- OID 1.3.6.1.4.1.25623.1.0.154909 - Apache HTTP Server < 2.4.64 Multiple Vulnerabilities - Linux: Apache mit dieser Version sind verwundbar für gleich mehrere Verwundbarkeiten
- OID 1.3.6.1.4.1.25623.1.0.104597 - Apache HTTP Server 2.4.0 - 2.4.55 HTTP Request Smuggling Vulnerability - Linux: Zeigt die Verwundbarkeit des Apache Webservers für eine sogenannte HTTP Request Smuggling-Schwachstelle an.
- OID 1.3.6.1.4.1.25623.1.0.125567 Apache HTTP Server 2.4.7 - 2.4.65 Authentication Bypass Vulnerability - Linux: Zeigt eine potentielle Schwachstelle an, mit der die Authentifizierung umgangen werden könnte.

5.3.6 NET3.1 Router und Switches

Router und Switches stellen eine bedeutende Säule der Datensammlung dar. Es werden Daten anhand der Ziel-IP-Adresse übermittelt. Da ein Nicht-Funktionieren ebendieser Geräte eine Beeinträchtigung für die gesamte IT-Infrastruktur bedeuten könnte, ist es wichtig diese gut abzusichern. Aufgrund der oftmaligen Verschmelzung diverser Funktionen von Routern und Switches kann der Großteil auf beide Geräte angewendet werden. Zur Überprüfung zweier Anforderungen dieses Bausteins wurden NVTs gewählt, die bereits in vorherigen Bausteinen Anwendung gefunden haben:

- Durch Grundschutz-Konfiguration
  - keine passende Einstellung vorhanden
- Durch Auswahl geeigneter NVTs
  - NET.3.1.A1 Sichere Grundkonfiguration eines Routers oder Switches
  - NET.3.1.A4 Schutz der Administrationsschnittstellen

NET.3.1.A1 Sichere Grundkonfiguration eines Routers oder Switches Wie auch Webserver müssen Router und Switches vor der Verwendung in sicherer Umgebung konfiguriert werden. Hierbei ist darauf zu achten, dass jegliche nicht notwendige Funktionen und Prozesse deaktiviert werden. Passwörter müssen vor ihrer Speicherung entsprechend verschlüsselt werden.

- OID 1.3.6.1.4.1.25623.1.0.170204 UPnP Detection (TCP): Zeigt, ob das Universal Plug and Play Protokoll aktiviert ist
- OID 1.3.6.1.4.1.25623.1.0.80091 TCP Timestamps Information Disclosure: Zeigt den Einsatz und Implementierung des TCP Timestamps, dadurch könnten automatische Berechnungen zur Uptime des Geräts unter Umständen berechnet werden

NET.3.1.A4 Schutz der Administrationsschnittstellen Administrations- und Managementzugängen sollten auf spezielle Bereiche beschränkt und durch eine separate Firewall geschützt werden. Auch hier gilt, dass nicht benötigte Bereich deaktiviert werden sollten.

- OID 1.3.6.1.4.1.25623.1.0.105610 - Weak MAC Algorithm(s) Supported (SSH): Zeigt, ob der Server schwache Client to Server Algorithmen unterstützt.
- OID 1.3.6.1.4.1.25623.1.0.105565 - SSH Protocol Algorithms Supported: Zeigt unterschiedliche Algorithmen, die vom SSH Service unterstützt werden
- OID 1.3.6.1.4.1.25623.1.0.105879 - SSL/TLS: HTTP Strict Transport Security (HSTS) Missing: Zeigt, ob der Server HTTP Strict Transport Security (HSTS) erzwingt

5.4 Prozessdiagramm

Als Grundlage für den im nächsten Kapitel präsentierten Prototypen dient das in Abbildung 5.3 dargestellte BPMN-Diagramm. Dieses veranschaulicht den System-Check-Prozess auf einer High-Level-Ebene und beschreibt die wesentlichen Schritte zur Identifikation und Bewertung von Systemen innerhalb eines Netzwerks.

Der Prozess beginnt mit der Initialisierung durch den Benutzer, der über eine Webanwendung auf das System zugreift. Im Anschluss erfolgt die Durchführung eines Host-Scans, dessen Ziel es ist, alle im Netzwerk erreichbaren Systeme zu identifizieren. Die erkannten Hosts werden anschließend extrahiert und zur weiteren Verarbeitung gespeichert.

Sofern Hosts identifiziert werden konnten, erfolgt eine detaillierte Analyse der einzelnen Systeme. Hierbei wird für jeden Host ein gezielter System-Scan durchgeführt, bei dem die ermittelten Informationen mit den Anforderungen des IT-Grundschutz-Kompendiums abgeglichen werden. Ziel dieses Schrittes ist es, potenzielle Schwachstellen zu identifizieren und deren Relevanz im Kontext der jeweiligen Anforderungen zu bewerten.

Die Ergebnisse dieses Abgleichs werden anschließend strukturiert aufbereitet und in einer

Datenbank gespeichert. Dabei werden sowohl erkannte Schwachstellen als auch entsprechende Handlungsempfehlungen zur Behebung dokumentiert. Abschließend werden dem Benutzer die aggregierten Ergebnisse in übersichtlicher Form zur Verfügung gestellt, sodass eine fundierte Bewertung des Sicherheitsstatus des Netzwerks möglich ist.

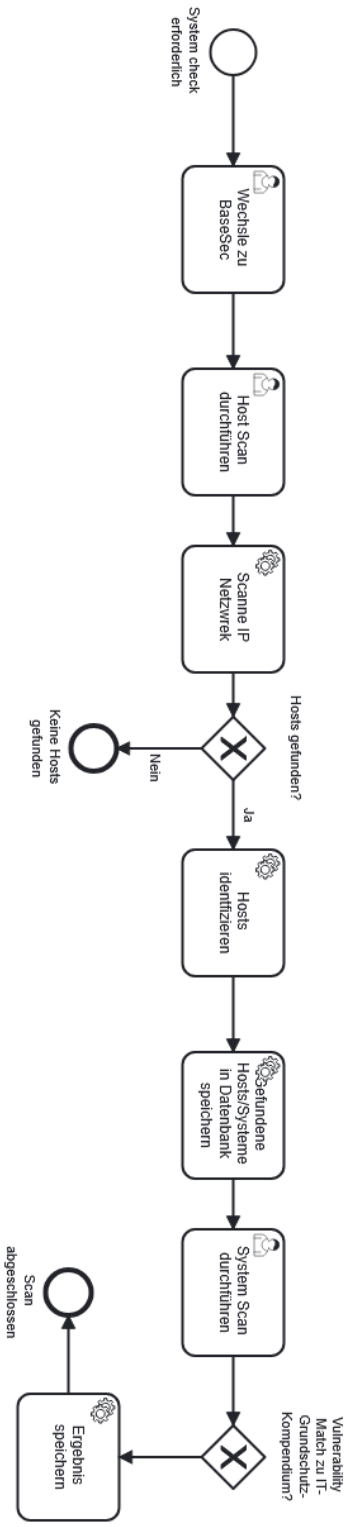

Abbildung 5.3: High-Level View

6 Prototyp

6.1 Ziele

Um einen funktionsfähigen Prototypen auf Basis des adaptierten Life-Cycle Modells zu realisieren, werden im Folgenden die Zielsetzung sowie die technischen Rahmenbedingungen definiert. Ziel ist die Implementierung einer Plattform, die Netzwerke scannt und identifizierte Hosts automatisiert anhand der in Kapitel 5 definierten Bausteine des IT-Grundschutz-Kompendiums bewertet. Der Test des Prototypen wird im nächsten Kapitel unter Einbeziehung verschiedener Testcases durchgeführt. Folgend möchte ich die Vorgehensweise und eingesetzten Technologien für die Implementierung des Prototyps erläutern.

Der erste Schritt besteht in der Ableitung eines Datenbank-Modells basierend auf dem entwickelten Modell aus dem Kapitel 5. Anschließend gilt es eine passende Entwicklungsumgebung und -sprache für die Plattform festzulegen. Durch den Einsatz eines Sequenzdiagramms wird die Interaktion der Implementierung und OpenVAS dargestellt. Dieses gilt es dann umzusetzen. Dafür müssen neben der Implementierung der Plattform, die Skripte entwickelt werden, die die Befehlskette für OpenVAS definieren. Diese Skripte werden mit dem GMP-Protokoll (Kapitel 3.7) umgesetzt und schließlich durch Interaktion des Users mit der Plattform ausgelöst. Zusammenfassend besteht der Prototyp demnach aus folgenden Komponenten:

1. Plattform
2. OpenVAS
3. Python Skripte - GMP-Protokoll

Im nachstehenden Abschnitt wird nun die Architektur des Prototypen erläutert und anschließend die Implementierung der Plattform beschrieben.

6.2 Architekturbeschreibung

Die Zusammensetzung und Kommunikation der unterschiedlichen Komponenten werden durch die nachfolgenden zwei Diagramme dargestellt.

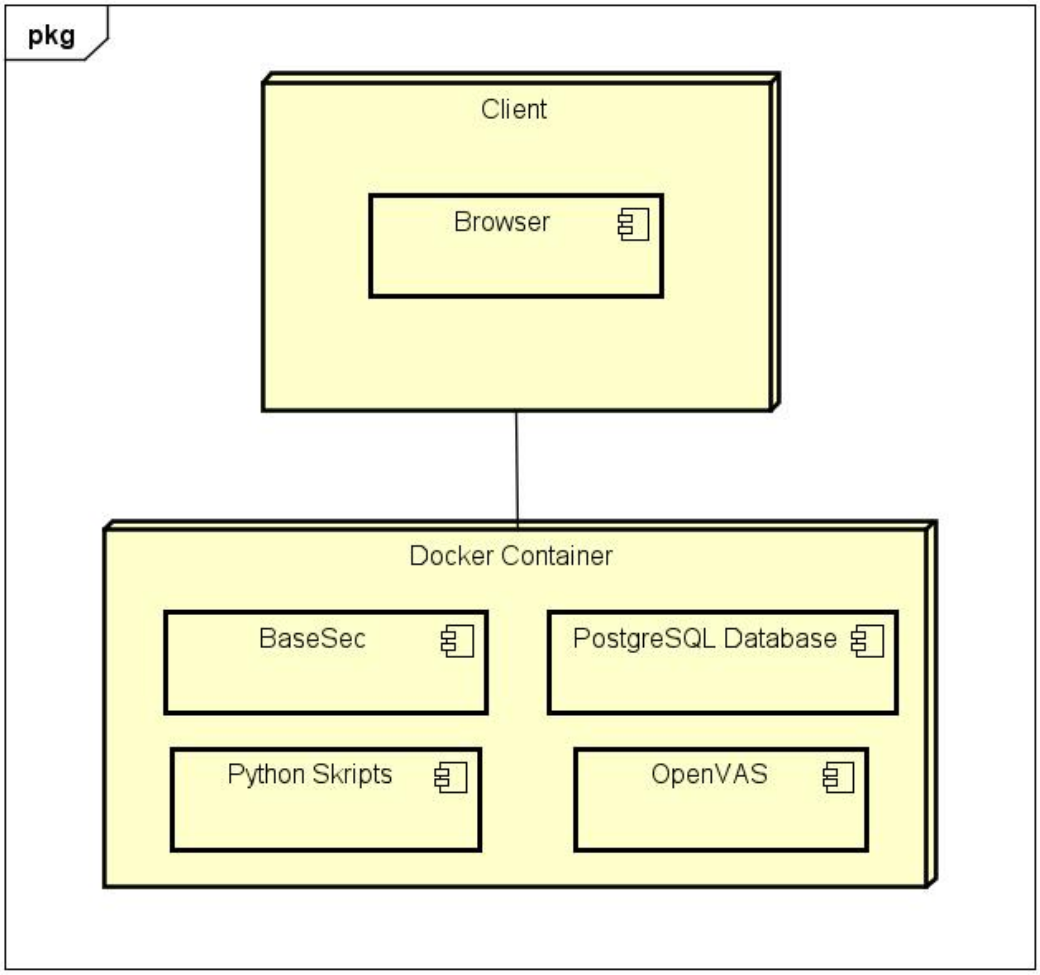

Abbildung 6.1: Bereitstellungsdiagramm

Das Bereitstellungsdiagramm in Abbildung 6.1 zeigt die Architektur und Zusammensetzung des entwickelten Prototyps sowie die Interaktion der einzelnen Komponenten.

Zentrales Element ist ein Docker-Container. Dieser Container umfasst sämtliche für den Betrieb des Prototyps notwendige Komponenten und stellt somit eine gekapselte und reproduzierbare Ausführungsumgebung dar. Innerhalb des Containers befinden sich mehrere funktionale Bestandteile:

- **BaseSec-Plattform:** Diese bildet die zentrale Anwendungsschicht und dient als Schnittstelle für den Benutzer. Über eine Weboberfläche kann der Benutzer Scanvorgänge initiieren, konfigurieren und die Ergebnisse einsehen.
- **OpenVAS:** Der OpenVAS-Scanner übernimmt die eigentliche Durchführung der Schwachstellenanalysen. Er wird durch die Plattform gesteuert und liefert strukturierte Scan-Ergebnisse zurück.

- **Python-Skripte:** Diese fungieren als Vermittlungsschicht zwischen der Plattform und OpenVAS. Über das GMP-Protokoll werden Scanbefehle generiert, ausgeführt und die Ergebnisse verarbeitet.
- **PostgreSQL-Datenbank:** Die Datenbank dient zur persistenten Speicherung von Scanergebnissen, identifizierten Schwachstellen sowie der zugeordneten Anforderungen des IT-Grundschutz-Kompendiums.

Die Interaktion zwischen den Komponenten erfolgt folgendermaßen: Der Benutzer greift über einen Webbrowser auf die Plattform zu und startet einen Scanprozess. Die Plattform übergibt die entsprechenden Befehle an die Python-Skripte, welche diese über das GMP-Protokoll an OpenVAS weiterleiten. Nach Abschluss des Scans werden die Ergebnisse zurückgegeben, aufbereitet und in der Datenbank gespeichert. Anschließend werden die Ergebnisse dem Benutzer in aggregierter Form dargestellt.

Durch die Containerisierung wird eine einfache Bereitstellung sowie eine klare Trennung der Komponenten erreicht. Gleichzeitig ermöglicht dieser Ansatz eine flexible Skalierung und erleichtert die Wartung des Systems.

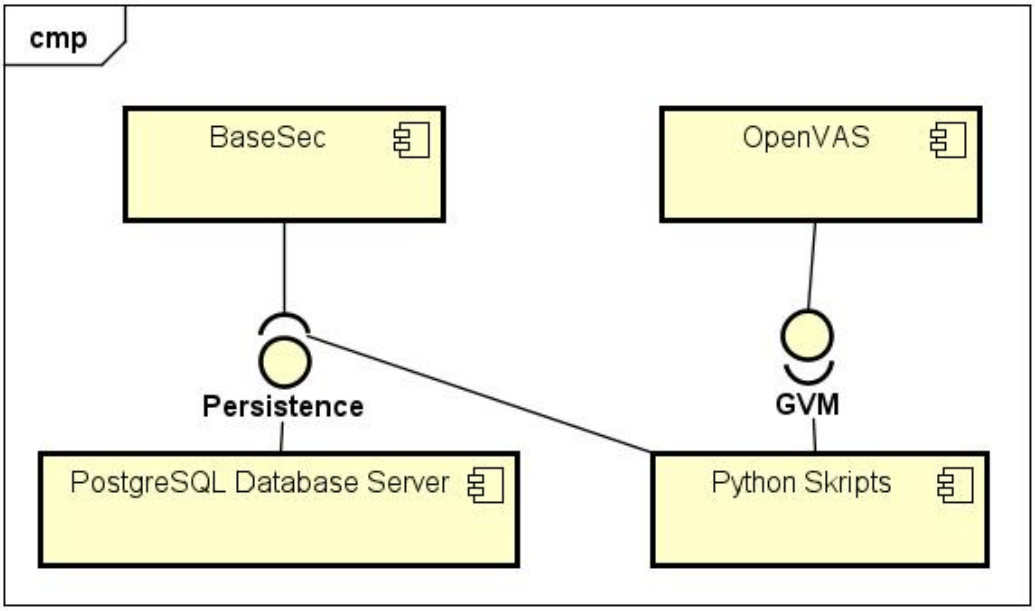

Abbildung 6.2: Komponentendiagramm

Das Komponentendiagramm in Abbildung 6.2 veranschaulicht die Kommunikationsbeziehungen zwischen den zentralen Komponenten des Prototyps sowie deren funktionale Abhängigkeiten.

Die BaseSec-Plattform stellt die zentrale Steuerungseinheit dar und interagiert sowohl

mit der Persistenzschicht als auch mit den Python-Skripten. Über die definierte Persistenzschnittstelle werden relevante Daten, wie Scanergebnisse und zugehörige Metadaten, in der PostgreSQL-Datenbank gespeichert und bei Bedarf wieder abgerufen.

Die Kommunikation zwischen den Python-Skripten und OpenVAS erfolgt über das Greenbone Management Protocol (GMP), welches durch das Greenbone Vulnerability Management (GVM) bereitgestellt wird. Die Python-Skripte übernehmen hierbei die Rolle einer Vermittlungsschicht, indem sie die von der Plattform initiierten Scanaufträge in entsprechende GMP-Befehle übersetzen und an OpenVAS übermitteln.

Nach Abschluss eines Scanvorgangs werden die Ergebnisse von OpenVAS über dieselbe Schnittstelle zurückgegeben und durch die Python-Skripte verarbeitet. Anschließend erfolgt die Weiterleitung der aufbereiteten Daten an die BaseSec-Plattform sowie deren Speicherung in der Datenbank.

Durch diese klare Trennung der Komponenten wird eine lose Kopplung zwischen den einzelnen Systembestandteilen erreicht, wodurch sowohl die Erweiterbarkeit als auch die Wartbarkeit des Prototyps verbessert werden.

6.3 Plattform - BaseSec

Aufgrund bisheriger Erfahrungen und Projekte fiel die Entscheidung auf eine REST Anwendung basierend auf Java. Zum Einsatz kommt dabei Dropwizard [15], ein Framework zur Implementierung von REST Services. Dabei setzt Dropwizard auf gängige Bibliotheken wie Jetty, Jersey und Jackson. Als grafische Oberfläche wurde zusätzlich ein Template basierend auf Bootstrap[14] verwendet. Platziert wird die Plattform auf demselben Server, der auch den OpenVAS Scanner beinhaltet. Dabei wird der Scanner verwendet, der in Kapitel 3.5 installiert und konfiguriert wurde. Als ersten Schritt für die Implementierung der Plattform muss der Server für das Framework vorbereitet werden. Die Voraussetzungen liegen dabei in der Installation von Java8 und Maven. Danach folgt die Grundkonfiguration von Dropwizard gemäß der Anleitung auf [15]. Nun wird das Klassendiagramm erläutert, welches in die Plattform integriert wird.

6.3.1 Klassendesign

Credential Diese Klasse enthält die Zugangsdaten, die der Benutzer über das Frontend für ein Netzwerk definieren kann. Die Zuordnung erfolgt bewusst auf Netzwerkebene und nicht auf Systemebene, um eine bessere Skalierbarkeit zu gewährleisten. In typischen Unternehmensumgebungen können so einheitliche Zugangsdaten für mehrere Systeme genutzt werden, wodurch der Verwaltungsaufwand reduziert wird.

Network Diese Klasse dient der logischen Gruppierung mehrerer Systeme. Sie ermöglicht eine strukturierte Verwaltung von Scan-Zielen und bildet die Grundlage für die Organisation

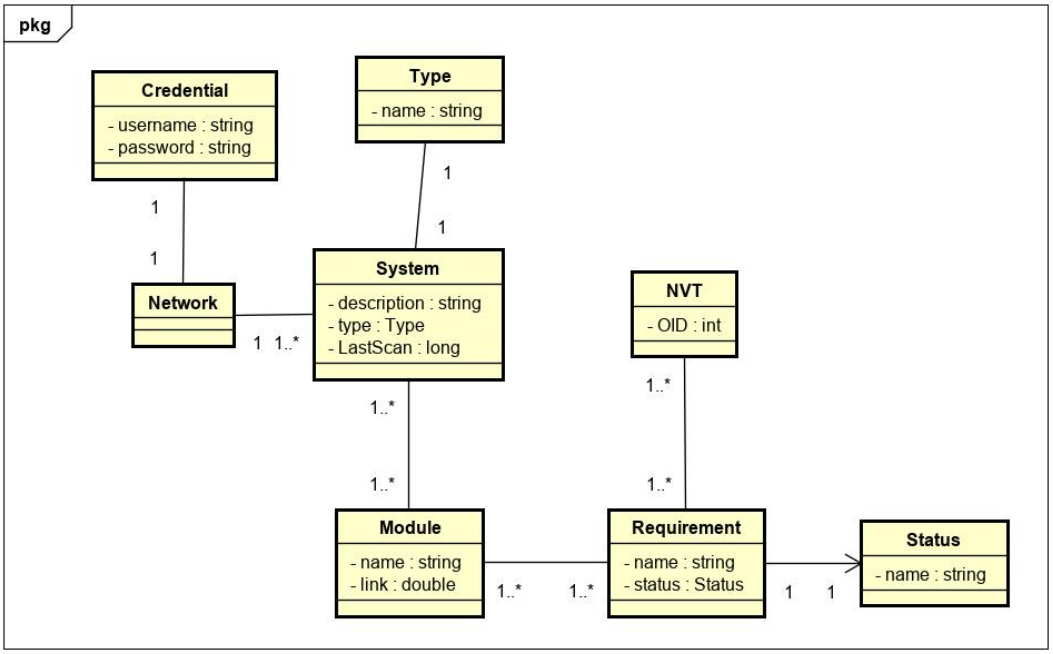

Abbildung 6.3: Klassendiagramm

unterschiedlicher Netzwerke innerhalb der Plattform.

Type Instanzen dieser Klasse repräsentieren den Typ eines Systems, beispielsweise Server, Client oder Netzwerkkomponente. Dadurch wird eine differenzierte Betrachtung und Auswertung der Systeme ermöglicht.

System Diese Klasse repräsentiert ein durch OpenVAS identifiziertes System. Neben einer Beschreibung werden der Systemtyp sowie der Zeitpunkt des letzten Scans gespeichert. Ein System kann mehreren Modulen zugeordnet sein, wodurch eine flexible Abbildung der Anforderungen des IT-Grundschutz-Kompendiums ermöglicht wird.

Module Diese Klasse bildet die im Prototypen berücksichtigten Bausteine des IT-Grundschutz-Kompendiums ab. Sie stellt eine zentrale Verbindung zwischen den identifizierten Systemen und den zugehörigen Anforderungen dar.

Requirement Diese Klasse repräsentiert einzelne Anforderungen eines Bausteins. Jede Anforderung ist einem Modul zugeordnet und besitzt zusätzlich einen Status, der den Erfüllungsgrad beschreibt.

Status Diese Klasse definiert die möglichen Zustände einer Anforderung, wie beispielsweise „erfüllt“, „nicht erfüllt“ oder „nicht automatisiert prüfbar“. Dadurch wird eine einheitliche Bewertung der Ergebnisse sichergestellt.

NVT Diese Klasse bildet die verwendeten Network Vulnerability Tests ab und stellt die Verbindung zwischen technischen Prüfungen und den fachlichen Anforderungen her. Eine Anforderung kann dabei durch mehrere NVTs überprüft werden, während ein NVT gleichzeitig für mehrere Anforderungen relevant sein kann.

Die Beziehungen zwischen den Klassen ermöglichen eine flexible und erweiterbare Modellierung des Gesamtsystems. Insbesondere die Zuordnung von NVTs zu Anforderungen sowie von Anforderungen zu Modulen stellt die zentrale Grundlage für das Mapping der Scan-Ergebnisse auf die Anforderungen des IT-Grundschutz-Kompendiums dar.

6.3.2 Sequenzdiagramm Scan-Prozess

Als Basis für die Umsetzung der jeweiligen Funktionen wird das folgende Sequenzdiagramm verwendet, das die Vorgehensweise und die Interaktion der unterschiedlichen Komponenten erläutert.

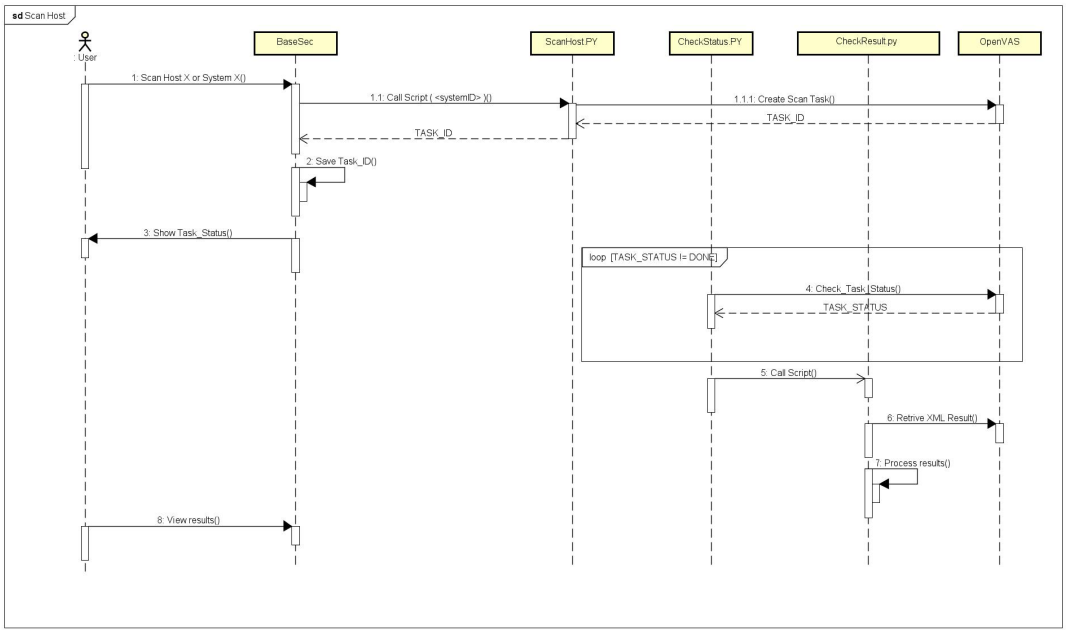

Abbildung 6.4: Sequenzdiagramm

Erläuterung Sequenzdiagramm:

1. Der Benutzer startet den Prozess, indem er die Funktion "Scan Host X or System X()" aufruft.

2. Der Prototyp "BaseSec" ruft das Skript "ScanHost.py" mit der entsprechenden Parametern auf.

3. "ScanHost.py" erstellt in OpenVAS einen neuen Scan-Task ("Create Scan Task()").
4. OpenVAS gibt eine eindeutige "TASK_ID" zurück, die an "ScanHost.py" und anschließend an "BaseSec" übermittelt wird.
5. "BaseSec" speichert die erhaltene "TASK_ID" lokal.
6. Dem Benutzer wird der aktuelle Status des Scan-Tasks angezeigt.
7. Solange der Scan-Task noch nicht abgeschlossen ist ("TASK_STATUS $\neq$ DONE"), wird in einer Schleife der Status abgefragt:
   a) Das Skript "CheckStatus.py" fragt den aktuellen Status des Tasks bei OpenVAS ab.
   b) OpenVAS liefert den aktuellen "TASK_STATUS" zurück.
8. Sobald der Scan abgeschlossen ist, ruft "CheckStatus.py" das Skript "CheckResult.py" auf.
9. "CheckResult.py" ruft die Scan-Ergebnisse im XML-Format von OpenVAS ab.
10. Die XML-Ergebnisse werden verarbeitet und analysiert.
11. Der Benutzer kann abschließend die aufbereiteten Scan-Ergebnisse einsehen.

6.3.3 Implementierung

Der Start der Implementierung erfolgt auf Basis des von Dropwizard zur Verfügung gestellten Beispiels [16]. Zunächst wurden die Schritte durchgeführt, damit das Beispiel im Tutorial lauffähig wird. Diese Version gilt als Basis für die Umsetzung des Prototyps. Anschließend wurden die Klassen erstellt, die im Klassendiagramm (Abbildung 6.3) vorgestellt wurden. Für die Abbildung der Daten wird PostgreSQL verwendet. Basierend auf dem REST Paradigma sind die Pfade nach folgendem Schema aufgebaut:

GET / :
Dieser Pfad liefert die Übersicht über gespeicherte Netzwerke

GET /network/{networkid} :
Hier erhält man eine Übersicht über die Systeme in diesem Netzwerk

GET /network/{networkid}/system/{systemId} :
Dieser Pfad liest die gespeicherten Modulanforderungen für dieses System aus und setzt Zählervariablen für umgesetzte und noch zu umsetzende Bausteine.

GET /network/{networkid}/system/{systemId} :
Detaillierte Informationen und Module eines spezifischen Systems können über diesen Pfad abgerufen werden.

GET /network/{networkId}/system/{systemId}/module/{moduleId}:
Alle implementierten Voraussetzungen sowie Detailinformationen zu einem bestimmten Modul eines Systems können hier eingesehen werden.

POST /network/{networkId}/startscan :
Durch das Aufrufen dieses Pfades wird das Python Skript "ScanHost.py" gestart, welches die verfügbaren Hosts im angegebenen Netzwerks scannt und in die Datenbank importiert. Außerdem wird es auch dazu verwendet, den Scan eines Systems zu initiieren. Eine genauere Erläuterung sowie ein Pseudocode befindet sich in Kapitel 6.3.4.

POST /network/{networkId}/system/{systemId}/startscan :
Hier wird ebenfalls das Skript ScanHost.py ausgeführt, allerdings wird der Parameter "System-Typ" mit "System" übergeben. Somit kann ein Scan für ein System ausgeführt werden.

GET /baustein/bausteinname :
Diese Ressource liefert bei Übergabe eines Bausteins die Informationen zu dessen zurück. Dafür wird das XML des IT-Grundschutz-Kompendiums maschinell verarbeitet und die relevanten Informationen extrahiert.

GET /anforderung/anforderungname :
Diese Ressource liefert bei Übergabe einer Anforderung die genaue Beschreibung zurück. Auch hierfür wird das XML des IT-Grundschutz-Kompenium maschinell verarbeitet und die Anforderung extrahiert.

Ergebnisverwertung Scan
Das zuvor genannte Python Skript startet die Ausführung des Scans über die Benutzeroberfläche von BaseSec. Um den Prozess zu überwachen und bei Fertigstellung entsprechende Maßnahmen zu treffen, existiert das Skript "CheckStatus.py", welches durch einen Cronjob in einem definierten Intervall, zum Beispiel jede Stunde, die laufenden Tasks bei OpenVAS überprüft und bei Fertigstellung das Skript "CheckResult.py" auslösen kann. Folgend nun die Pseudocodes, die die Funktionsweise der jeweiligen Skripts näher erläutern.

6.3.4 Python-Skripte

Algorithm 1 ScanHost.py

```
1: Verbinde mit Datenbank basesec_db und öffne das Logfile "HostScan.txt"
2: if SystemTyp == System then
3:     Erstelle durch das GMP Protokoll den Task für das Scannen eines Systems auf dem OpenVAS System
4:     Update den Eintrag des Systems in der Datenbank mit der erhaltenen TaskID von OpenVAS
5: end if
6: if SystemTyp == Network then
7:     Erstelle durch das GMP Protokoll den Task für das Scannen des Netzwerks auf dem OpenVAS System
8:     Update den Eintrag des Systems in der Datenbank mit der erhaltenen TaskID von OpenVAS
9: end if
10: Logfile mit den Infos befüllen
```

Algorithm 2 CheckStatus.py

```
1: Verbinde mit Datenbank basesec_db"
2: while Tasks in Datenbank vorhanden do
3:     if Task von Datenbank == "Done" in OpenVAS then
4:         Update den Status in der Datenbank und LastScan
5:         if TaskType == Network then
6:             Triggere das Skript "CheckResult.py"
7:         end if
8:     end if
9: end while
10: Logfile mit den Infos befüllen
```

Algorithm 3 CheckResult.py

```
Verbinde mit Datenbank basesec_db
Hole aus der Datenbank die zugehörige TaskID für das System abhängig vom Systemtyp
Lade die Scan-Ergebnisse aus OpenVAS für diese TaskID
if systype == Network then
    Extrahiere alle gefundenen Hosts aus den Scan-Ergebnissen (IP + Name/Hostname)
    if nvt-name == OS Detection Consolidation and Reporting then
        Finde die Beschreibung und Scanne den Text nach OS
        Nutze eine Regex zum Extrahieren des gefundenen OS
    end if
    Aktualisiere/füge gefundene Hosts in der Datenbank ein
end if
if systype == System then
    Durchlaufe relevante Scan-Resultate und mappe sie auf Requirements (OID/NVT → Requirement)
    Berechne Status und Erklärung pro Requirement und update die Tabelle
    Prüfe zusätzlich IT-Grundschutz-Compliance-Resultate und update Requirements entsprechend
end if
```

6.3.5 Frontendübersicht

Als erster Screen (Abbildung 6.5) der Applikation dient ein Überblick über alle erfassten Netzwerke einschließlich der umgesetzten Anforderungen aller Systeme in diesem Netzwerk.

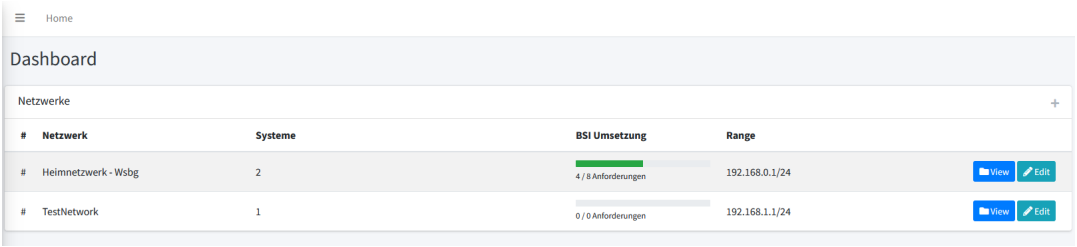

Abbildung 6.5: Dashboard

Auf der Detailseite jedes Netzwerks (Abbildung 6.6) werden sowohl die einzelnen Systeme dieses Netzwerks aufgelistet, als auch die Aktion zum Suchen von Hosts in diesem Netzwerk ausgelöst. Die genaue Funktionsweise wird im Kapitel 6 erläutert, welches einen Überblick über die eingesetzten Python-Skripte liefert.

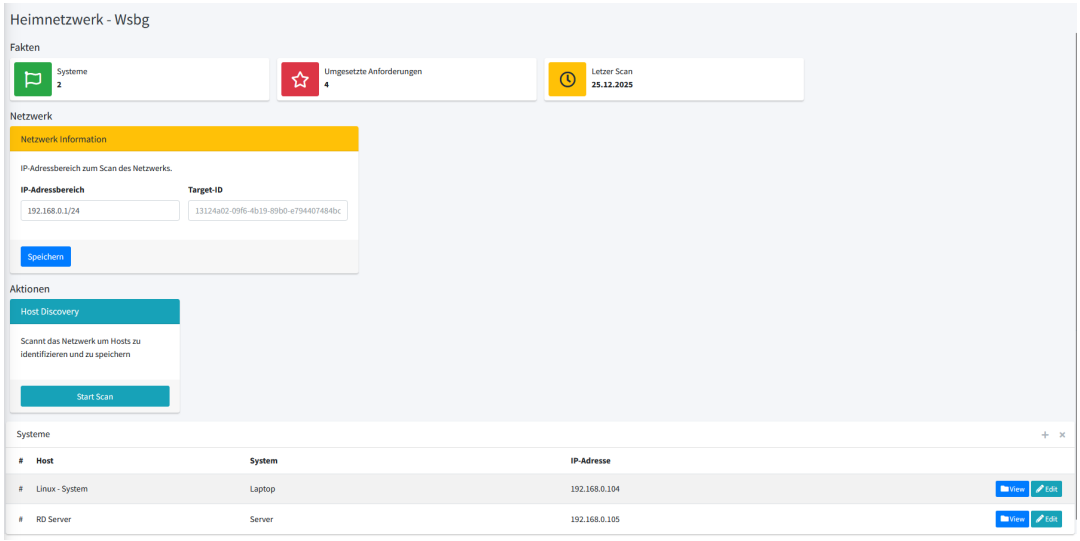

Abbildung 6.6: Netzwerk

Jedes System verfügt über einen Überblick über alle umgesetzten Anforderungen und die Auslösung des Scans (Abbildung 6.7). Zusätzlich kann die IP Adresse des Systems bearbeitet werden.

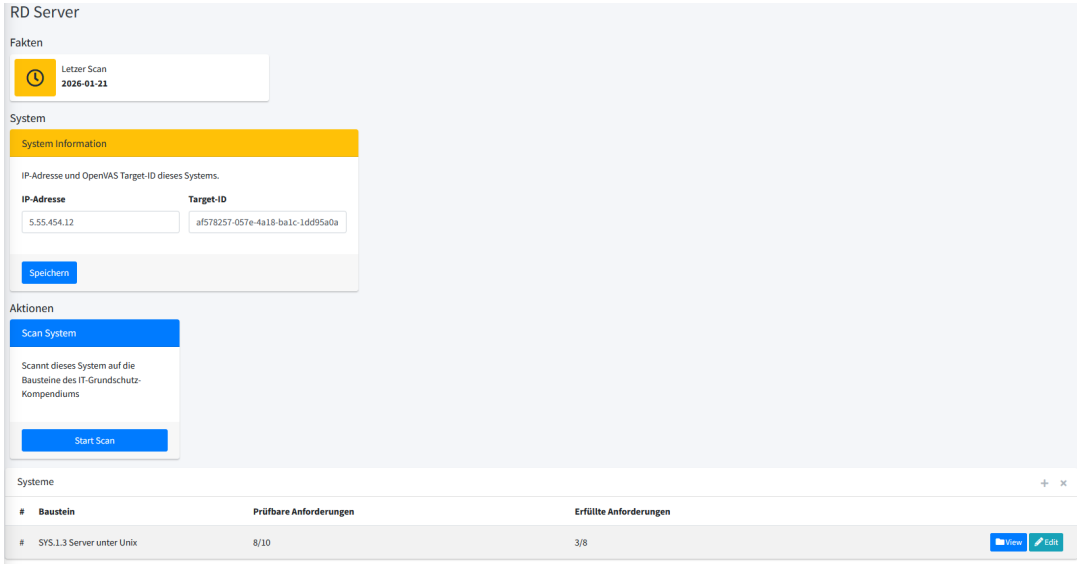

Abbildung 6.7: Dashboard

Abschließend gibt es für jedes System einen Überblick über die genauen Anforderungen und deren Status. Über einen Klick auf Detail wird das Ergebnis des Scans, diesen Baustein betreffend angezeigt (Abbildung 6.8 und 6.9).

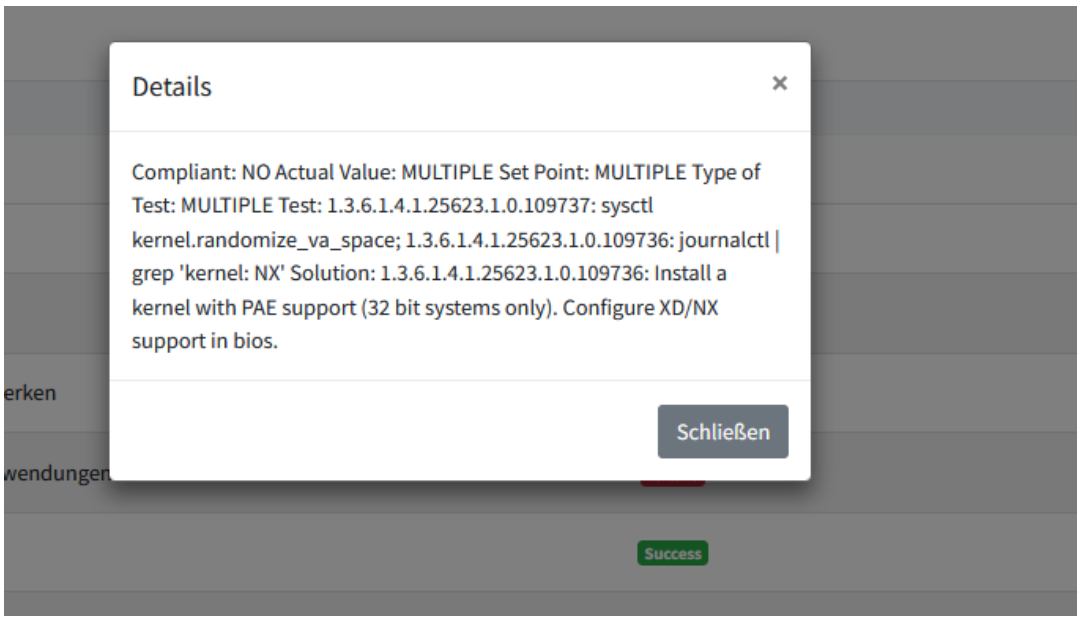

Abbildung 6.9: System Detailseite - Anforderung Detail

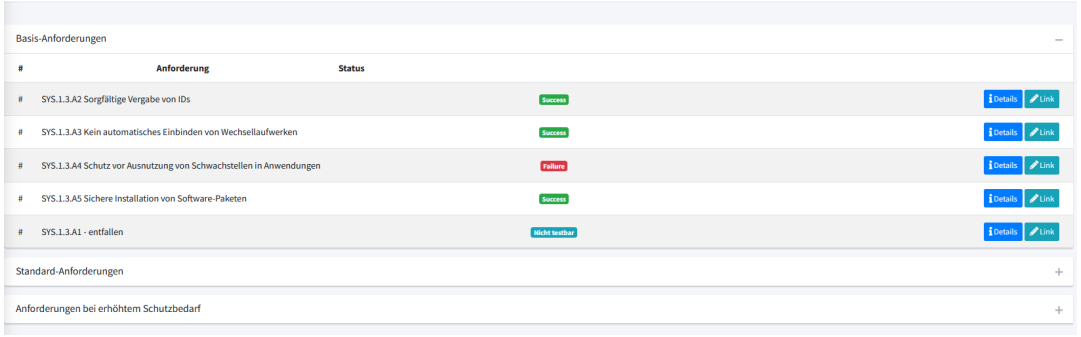

Abbildung 6.8: System Detailseite

7 Evaluation und Ergebnisse

Um die erörterten Probleme und umgesetzten Maßnahmen zu testen, werden in diesem Kapitel unterschiedliche Test Cases analysiert und ein Fazit aus den gewonnenen Erkenntnissen gezogen.

7.1 Test Case 1: Linux Server

Als erstes Testobjekt wurde ein Linux-Server verwendet, der als Webserver für unterschiedliche Applikationen dient. Dazu wurde ein authentifizierter Scan durchgeführt. Ziel des Tests war es das System gegen den Baustein SYS.1.3 Server unter Linux und Unix zu testen. Als Grundsetup wurden SSH Zugangsdaten in OpenVAS hinterlegt und das Zielsystem erstellt. Anschließend wurden die entsprechenden Anforderungen dieses Bausteins, unter Verwendung der in Kapitel 5 erläuterten Anforderungen für diesen Baustein, in BaseSec geprüft. Die Dauer des Scanvorgangs betrug hierbei knapp eineinhalb Stunden. Als Ergebnis konnten die folgenden Status ermittelt werden:

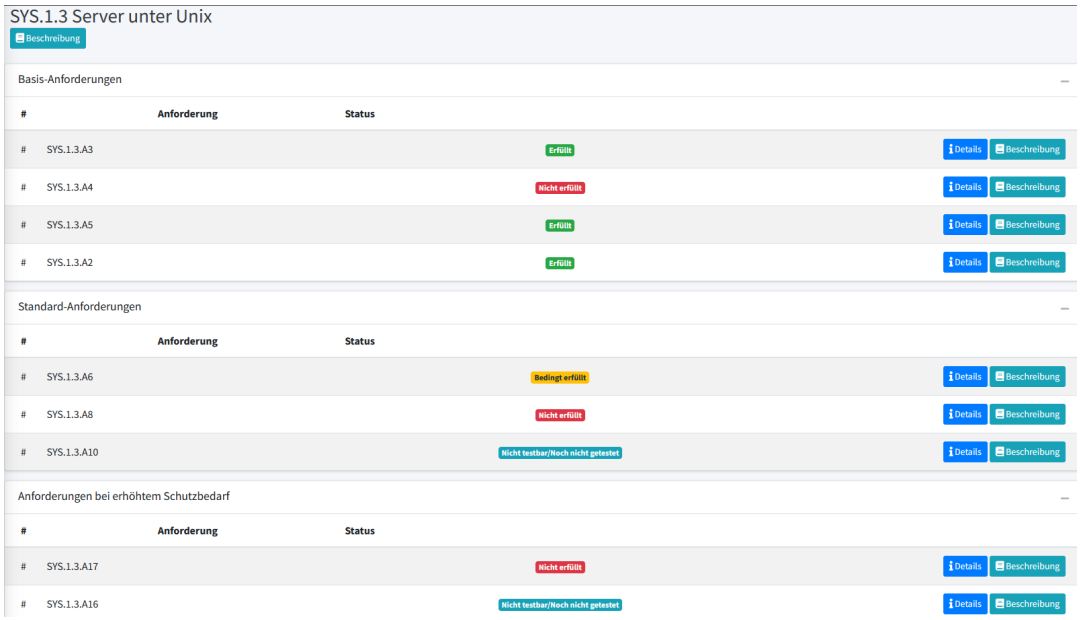

Abbildung 7.1: Linux Server Ergebnisse BaseSec

Von den acht ausgewählten Anforderungen dieses Bausteins konnten demnach drei Anforderungen als erfolgreich markiert werden. Bei den drei fehlgeschlagenen sowie dem einen be-

dingt erfüllten Ergebnis lassen sich in detaillierter Form mögliche Lösungsvorschläge anzeigen. So wird in Abbildung 7.1 und Abbildung 7.2 das Problem für die Validierung der Anforderung SYS.1.3.A8 Verschlüsselter Zugriff über Secure Shell erläutert mit dem Lösungshinweis die schwachen MAC Algorithmen zu deaktivieren.

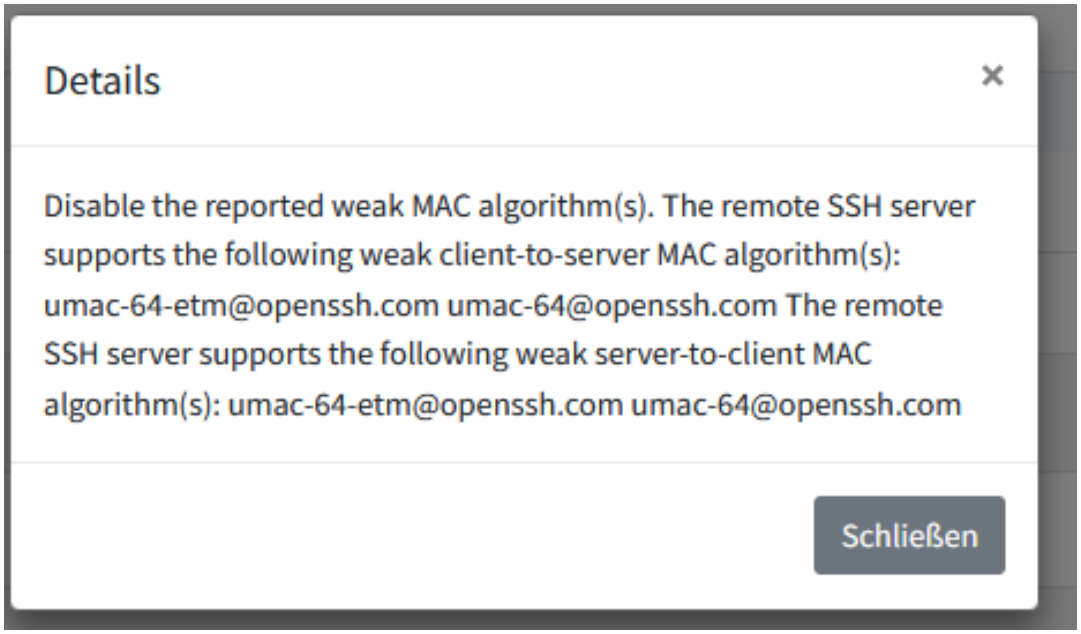

Abbildung 7.2: Linux Server Ergebnis Modal BaseSec

Die IT-Grundschutz-Kompendium Scankonfiguration bietet somit für die Basis-Anforderungen großteils automatisierte Tests. Lediglich für die Basis-Anforderung SYS.1.3.A5 steht in der Detailbeschreibung, dass es erfolgreich getestet wurde, aber eine manuelle Überprüfung notwendig ist. Angesichts der Anforderung für die Sichere Installation von Software-Paketen ist dies auch nachvollziehbar, wobei sich die Frage stellt, wie der Scan hier die erfolgreiche Prüfung dieser Anforderung feststellen konnte. Die restlichen Anforderungen dieses Bausteins beruhen auf Prinzipien, die eine automatische Feststellung großteils ausschließen. Eine manuelle Auswahl geeigneter NVT wie es hier gemacht wurde, um weitere Anforderungen zu testen, ist aufgrund der enormen Anzahl an NVT nicht realistisch. Dennoch bieten gerade solche Systeme einen optimalen Testeinstieg zur Überprüfung nach dem IT-Grundschutz-Kompendium, weil sehr viele Schwachstellen identifiziert werden, die vor allem für eine notwendige Grundsicherheit relevant sind. Dies beginnt bereits bei der Erkennung veralteter Software oder notwendigen Patches für installierte Pakete, die einen wichtigen Input für eine sichere Umgebung liefern.

7.2 Test Case 2: REST Applikation

Als zweites Testobjekt wurde eine REST Applikation auf JAVA Basis gewählt, um feststellen zu können, welche Schwächen beim Scannen hier gefunden werden. Diese Applikation stellt eine Custom Implementierung einer Webapplikation dar. Hierfür wurden die Bausteine APP.3.1.

Webanwendungen und Webservices, APP.3.2. Webserver und APP.4.3 Relationale Datenbanken ausgewählt. Dabei wurden alle NVT, wie in Kapitel 5 erläutert, für diese Bausteine manuell ausgewählt, weil hierfür keine automatisierten Konfigurationen existieren. Die Abbildung 7.3 und Abbildung 7.4 zeigen dabei die Ergebnisse nach dem Mapping. Aus den ausgewählten NVT konnte keine Anforderung eindeutig als erfüllt gekennzeichnet werden. Die Anforderung APP.3.2.A13 wird hier lediglich als "Bedingt erfüllt" gekennzeichnet, weil die ausgewählten NVT nicht im Ergebnis nachgewiesen werden konnten. Dies kann aber auch aus Time-Outs während des Testings stammen und gilt somit nicht als eindeutiger Faktor für die Erfüllung der Anforderung. Die Dauer des Scanvorgangs betrug hierbei knapp eineinhalb Stunden.

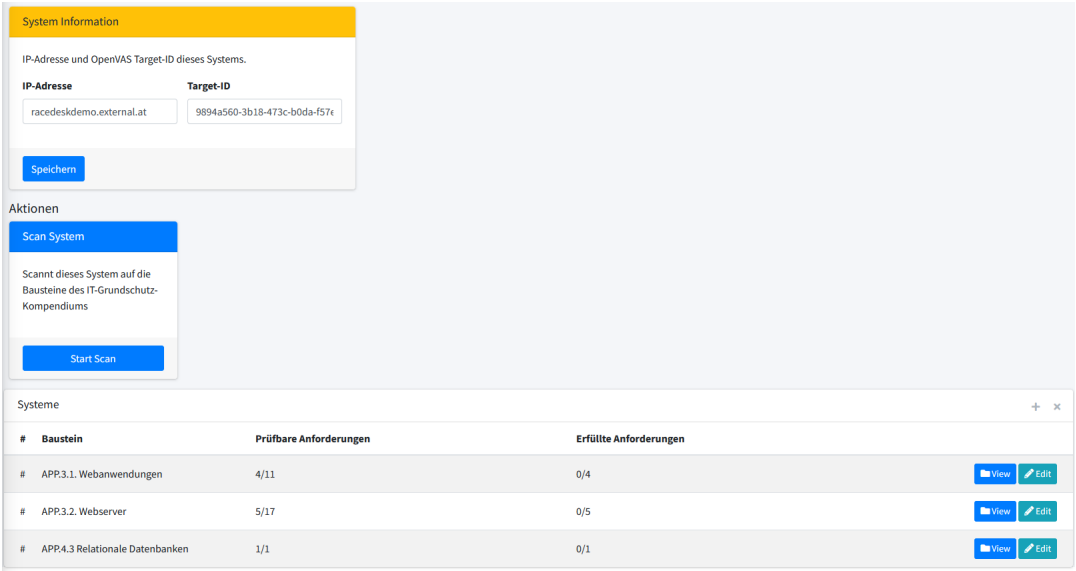

Abbildung 7.3: Java Webapplikation getestete Bausteine

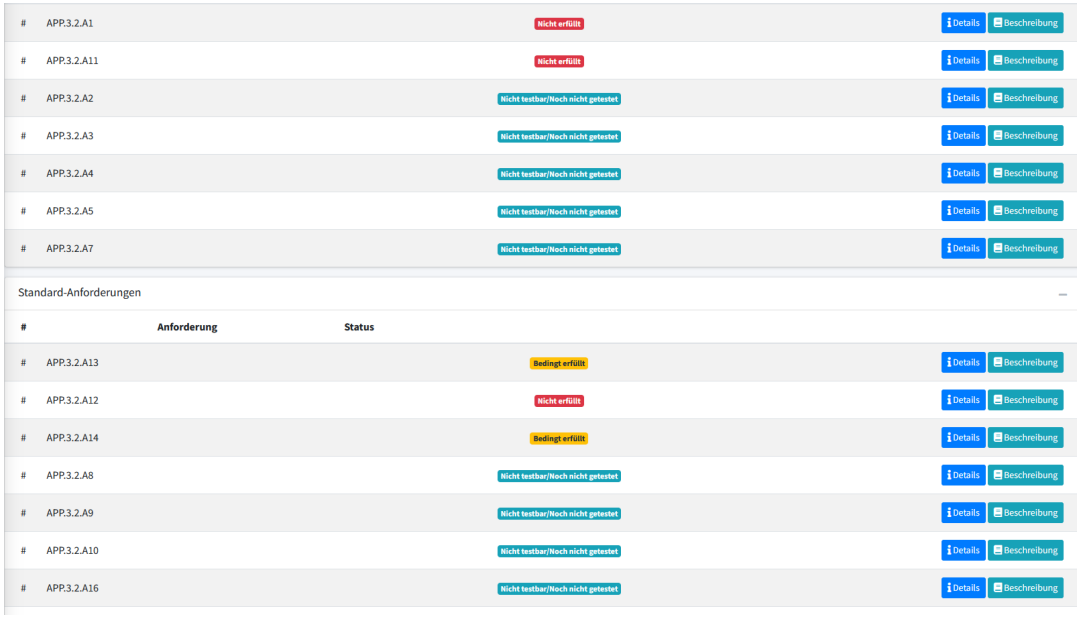

Abbildung 7.4: Java Webapplikation Detailansicht APP.3.2

7.3 Test Case 3: Wordpress Instanz

Test Case 3 stellt den Scan auf eine Website basierend auf Wordpress - einem CMS, das einen weltweit hohen Einsatz aufweist. Aufgrund des starken Wandels hinzu serviceorientierten Plattformen/Websites, die zahlreiche Kundendaten speichern und verarbeiten, spiegelt dies einen wichtigen Praxisfall zur Prüfung wider. Die ausgewählten Bausteine decken sich hier mit den Bausteinen aus Test Case 2. Die Tests, die hier von OPENVAS durchgeführt werden können, sind zahlreich und mündeten in 608 gefundenen Ergebnissen (Darunter auch viele Log-Informationen). Im Vergleich zur Custom REST-Applikation aus Test Case 2, bei der 181 Ergebnisse gefunden wurden, zeigt dies, dass vor allem auch für Open Source Systeme eine signifkant höhere Anzahl an Tests vorhanden ist. Daraus resultiert auch eine wesentlich längere Scandauer als bei den vorherigen beiden Tests. In Summe wurden hier fast fünf Stunden für die Prüfung der Applikation aufgebracht. Das Ergebnis wirkt auf den ersten Blick identisch zum Ergebnis in Test Case 2. Allerdings wurden hier zusätzliche beziehungsweise teilweise andere NVT gematcht, die zum Ergebnis führen. So sieht man, dass zum Beispiel hier die Anforderung APP.3.2.A13 als nicht erfolgreich markiert wurde, wohingegen APP.3.2.A11 hier als "Teilweise Erfüllt" gekennzeichnet wurde. Ein Detailblick (Abbildung 7.7 auf die Anforderung A11 zeigt zum Beispiel, dass die zwei entdeckten NVT der Grund für den angegebenen Status sind. Außerdem wird ein Lösungsvorschlag zur Behebung präsentiert.

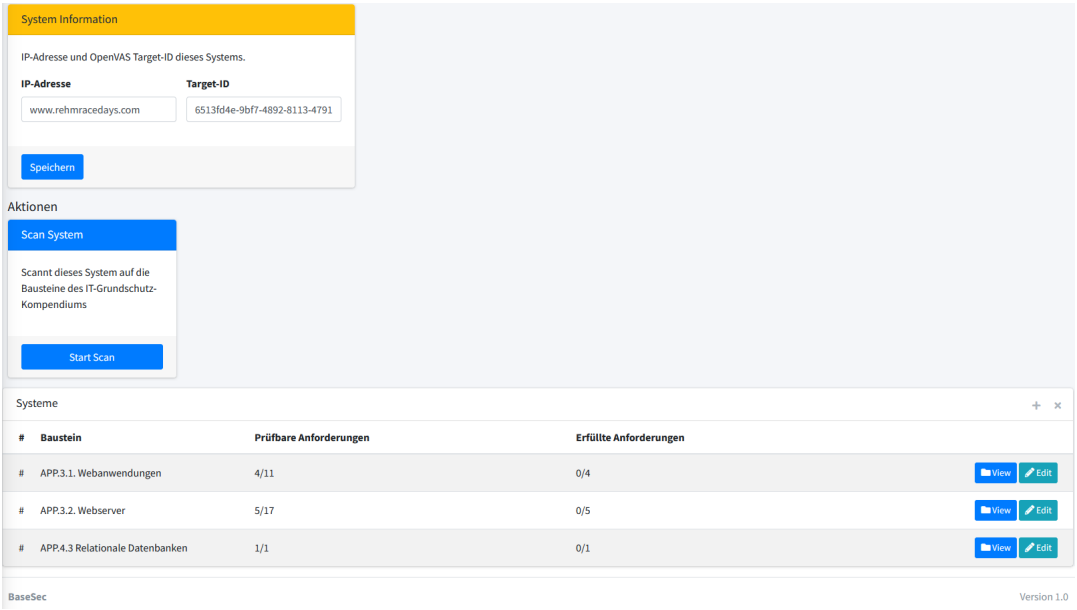

Abbildung 7.5: Wordpress Applikation getestete Bausteine

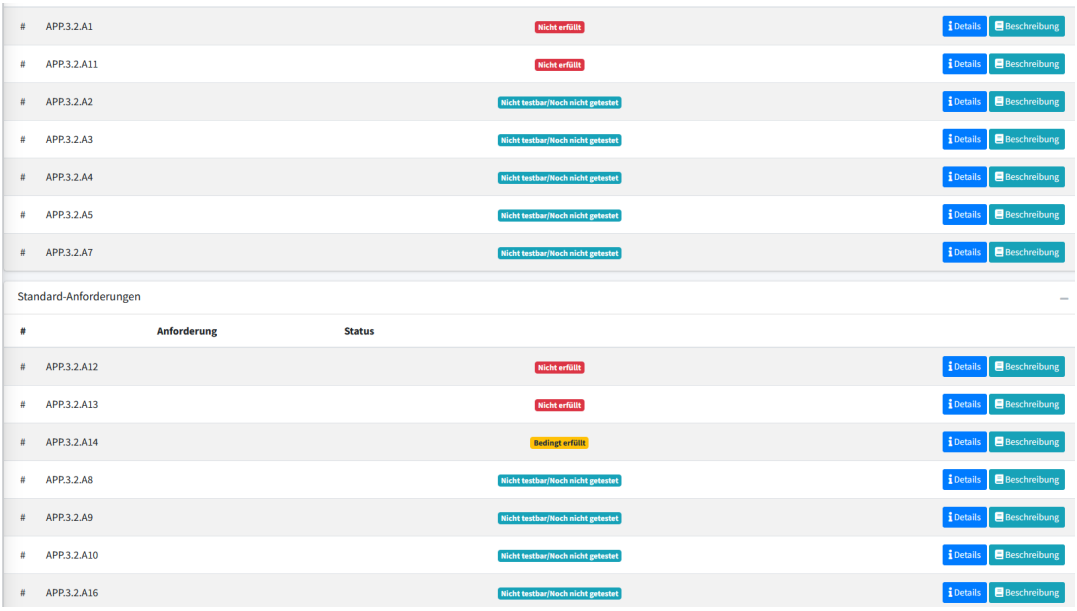

Abbildung 7.6: Wordpress Applikation Detailansicht APP.3.2

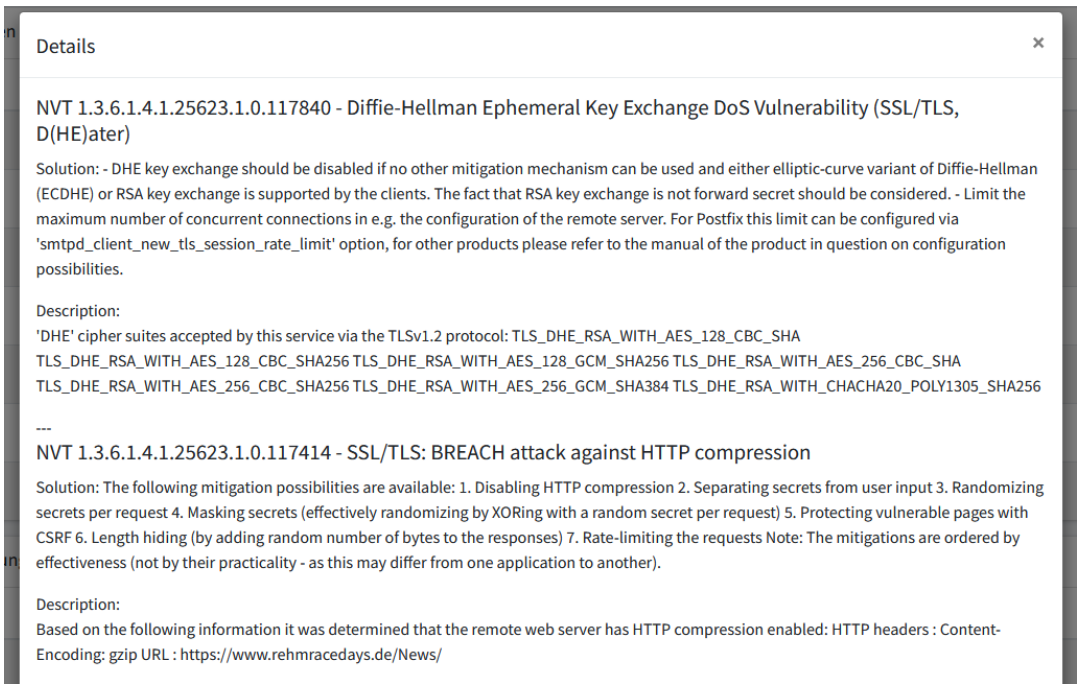

Abbildung 7.7: Wordpress Applikation Detailansicht APP3.2.A11

7.4 Test Case 4: Heimnetzwerk

Der letzte Test wurde im Heimnetzwerk durchgeführt. Hierfür wurde zuerst ein Host-Discovery angestoßen, um alle im Netzwerk befindlichen Geräte zu identifizieren. Das Python Skript "CheckResult.Py" hat daraufhin die gefundenen Hosts extrahiert und gemäß der eingesetzten Regex bestmöglich versucht auch einen lesbaren Namen aus dem gefundenen Host zu ziehen. Abbildung 7.8 zeigt das Ergebnis dieses Scans. Als nächsten Schritt habe ich gezielt ein System gewählt, das ich näher auf den Baustein NET.3.1 betrachte. Dafür habe ich den Router des Netzwerks gewählt.

#	Host	System	IP-Adresse	
#	sonos-542a1b5df270.local	Device	10.0.0.16	View Edit
#	QNAP	Device	10.0.0.11	View Edit
#	myfritz.box	Device	10.0.0.1	View Edit
#	Microsoft	Device	10.0.0.27	View Edit
#	10.0.0.24	Device	10.0.0.24	View Edit
#	Linux	Device	10.0.0.7	View Edit
#	Zyxel	Device	10.0.0.20	View Edit
#	10.0.0.110	Device	10.0.0.110	View Edit
#	10.0.0.69	Device	10.0.0.69	View Edit
#	10.0.0.9	Device	10.0.0.9	View Edit
#	10.0.0.220	Device	10.0.0.220	View Edit
#	10.0.0.221	Device	10.0.0.221	View Edit
#	10.0.0.28	Device	10.0.0.28	View Edit
#	10.0.0.111	Device	10.0.0.111	View Edit
#	10.0.0.150	Device	10.0.0.150	View Edit
#	10.0.0.23	Device	10.0.0.23	View Edit

Abbildung 7.8: Gefundene Hosts in Heimnetzwerk

Im Vergleich zu den anderen untersuchten Bausteinen, zeigt dieser, dass er über wesentlich mehr Anforderungen, vor allem auch im organisatorischen Kontext, verfügt (Abbildung 7.9 und Abbildung 7.10 . Ein genauer Blick auf die Abbildung 7.11 zeigt dabei den genauen Grund für die nicht erfüllte Anforderung. Darin sieht man zum Beispiel, dass UPnP erkannt wurde inklusive der genauen Produktspezifikation des Routers.

System

System Information

IP-Adresse und OpenVAS Target-ID dieses Systems.

IP-Adresse: 10.0.0.1

Target-ID: 1c386b38-67dc-44bb-80f8-88f2

Speichern

Aktionen

Scan System

Scannt dieses System auf die Bausteine des IT-Grundschutz-Kompendiums

Start Scan

Systeme

#	Baustein	Prüfbare Anforderungen	Erfüllte Anforderungen	
#	NET3.1 Router und Switches	2/26	0/2	View Edit

Abbildung 7.9: Fritzbox getestete Bausteine

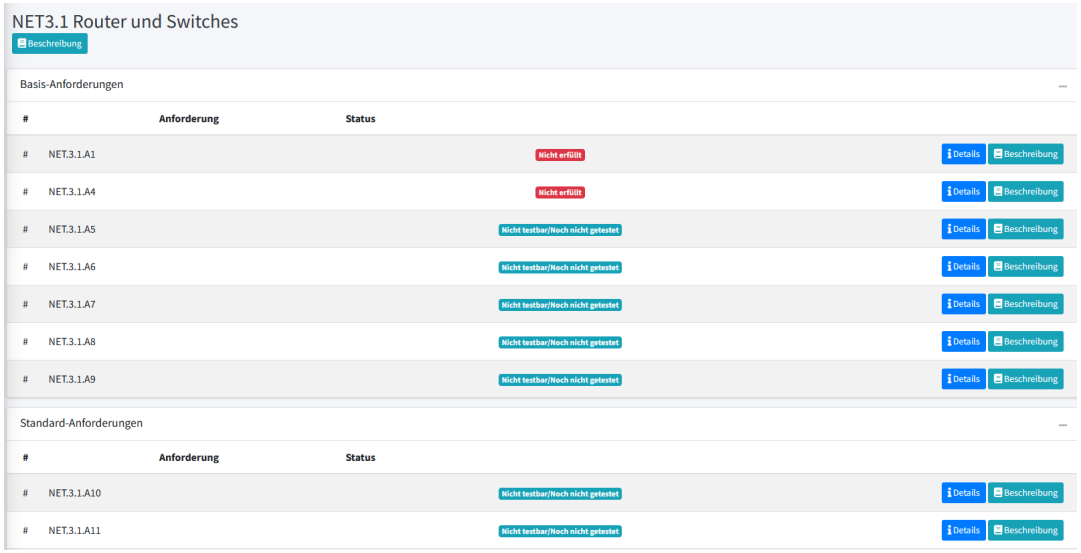

Abbildung 7.10: Fritzbox Detailansicht

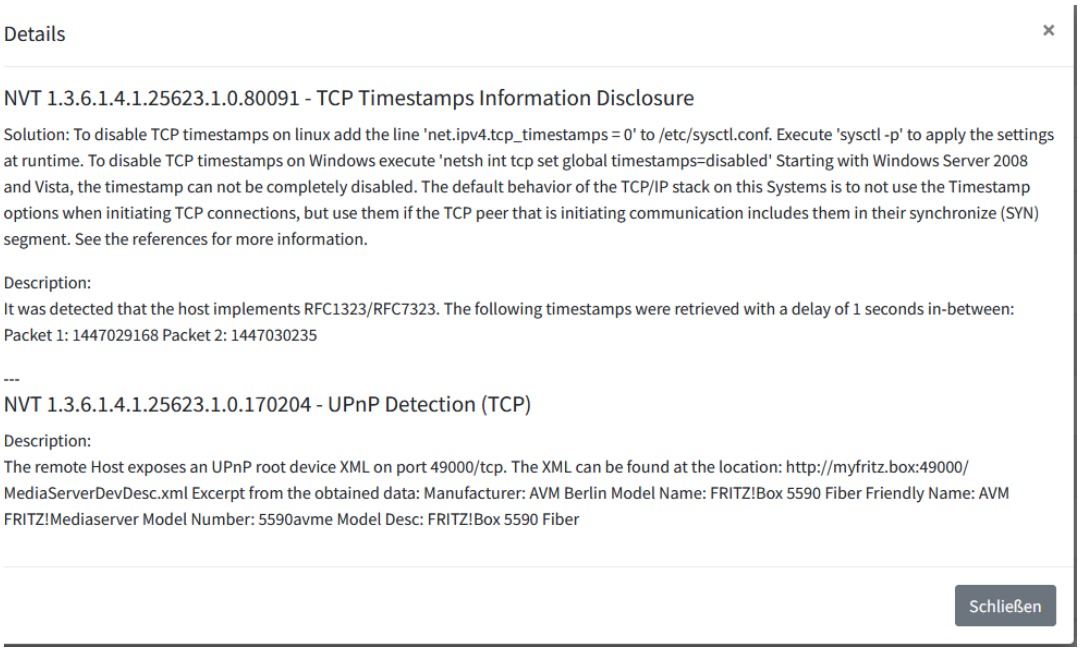

Abbildung 7.11: Fritzbox Detailansicht NET.3.1.A1

7.5 Ergebnisse

Die durchgeführten Testfälle zeigen, dass Teile getestet werden können, aber ein vollautomatisiertes System in vielen Fällen nicht möglich ist, weil jedes System eine individuelle Prüfung benötigt, die abhängig ist von ihrem Einsatzzweck. Das BSI IT Grundschutz Kompendium bie-

tet in vielen Punkten eine Design- bzw Architekturbeschreibung oder auch organisatorische Maßnahmen, die für automatisierte Tests nur beschränkt oder nicht testbar sind. So können Hilfstools wie in dieser Arbeit erläutert, einen Einstieg zur Prüfung verschiedener Systeme geben. Der überwiegende Großteil der Überprüfungen muss jedoch auf manuellem Weg erfolgen. Vor allem im Bereich von Servern wie im Test Case 1 gezeigt, kann die automatische Überprüfung einen ersten Überblick über die geforderten Anforderungen bringen. Der Host Discovery Scan wie im Test Case 2 erläutert, kann zudem einen nutzbringenden Einblick über alle im Netzwerk befindlichen Systeme zeigen und somit potentielle Sicherheitsgefährdungen aufzeigen. Der Einsatz diverser Unterstützungstools ist somit unabdingbar als Grundstein zur Umsetzung diverser Grundschutz Frameworks. Gleichzeitig zeigt es aber auch, vor allem durch den doch teilweise hohen Interpretationsspielraum, dass ein Standard zwar eine Richtung vorgeben, keineswegs aber einfach in ein Unternehmen integriert werden kann. Der vorgestellte Prototyp bietet eine Basis zur Identifizierung von Hosts als auch dem Testing festgelegter Bausteine und Anforderungen. In einem weiteren Ausbau könnte sowohl der Einsatz einer KI von Vorteil sein, sei es zum Mapping von Ergebnissen oder auch dem Vorschlag zur Auswahl neuer NVTs für das Mappen zu gewissen Anforderungen. Oder auch der Einsatz mehrerer Scanner, um die potentielle Suche noch weiter zu erhöhen. Außerdem könnte man auf ein System festgelegte eigene NVTs in Form von NASL Skripten implementieren, um speziell auf ein im Unternehmen relevantes System eine Anforderung zu testen. Ein weiterer nicht zu unterschätzender Punkt ist die Skalierung des Systems. Die durchgeführten Testfälle zeigen bei genaueren Scankonfigurationen, dass ein System eine erhebliche Zeit für die Durchführung der Prüfung beansprucht.

8 Conclusio

Im Zentrum dieser Arbeit stehen verschiedene Szenarien zur Identifikation von Schwachstellen unter Einsatz von OpenVAS in Kombination mit dem IT-Grundschutz-Kompendium.

Zu Beginn der Arbeit standen nur sehr wenige Werkzeuge zur Verfügung, die eine Unterstützung bei der Anwendung des IT-Grundschutzes ermöglichten. Inzwischen existiert eine größere Anzahl an Tools unterschiedlicher Anbieter, die bei der Umsetzung des Maßnahmenkatalogs unterstützen. Dennoch fehlt weiterhin eine automatisierte Möglichkeit zur schnellen Ersteinschätzung des Umsetzungsgrads aller relevanten Bausteine.Dies ist zum einen auf den hohen Anteil organisatorischer Anforderungen zurückzuführen, die sich grundsätzlich nur schwer automatisiert überprüfen lassen. Zum anderen erschwert die individuelle Struktur von IT-Systemen und Netzwerken eine standardisierte Bewertung. Die Ergebnisse des vorherigen Kapitels zeigen, dass zwar geeignete Methoden existieren, die eine fundierte Unterstützung bei der Bewertung einzelner Bausteine bieten, jedoch häufig spezifische Network Vulnerability Tests (NVTs) oder sogar individuell entwickelte NASL-Skripte erforderlich sind.

Die Herausforderung der automatisierten Informationsgewinnung aus dem IT-Grundschutz-Kompendium konnte im Rahmen dieser Arbeit durch den Einsatz von XPath in Kombination mit regulären Ausdrücken (RegEx) adressiert werden. Deutlich problematischer ist hingegen die Bewertung der Scan-Ergebnisse selbst. Viele von OpenVAS identifizierte Schwachstellen werden lediglich als *LOG*-Einträge klassifiziert und stellen aus Sicht des Scanners keine unmittelbare Bedrohung dar. Im Kontext des IT-Grundschutz-Kompendiums können diese Informationen jedoch sicherheitsrelevant sein, beispielsweise wenn Systeminformationen preisgegeben werden, die ein Ausspähen erleichtern.

Dies führt zu einer erhöhten Komplexität bei der Bewertung, da die Erfüllung einer Anforderung stark von der jeweiligen Interpretation sowie den spezifischen Rahmenbedingungen des betrachteten Systems abhängt.Eine wesentliche Verbesserung ist durch die Einführung des IT-Grundschutz++ zu erwarten, dessen Veröffentlichung für das Jahr 2026 angekündigt ist. Ziel dieses erweiterten Ansatzes ist es, die automatische Verarbeitung von Bausteinen, Anforderungen und Bewertungen durch den Einsatz standardisierter Satzschablonen zu erleichtern [11].

Darüber hinaus zeigt das Dokument [11] aus Oktober 2023, dass ein dediziertes IT-Grundschutz-Tool geplant ist, welches Automatisierungen sowie konkrete Handlungsvorschläge unterstützen soll. Ergänzend wurde kürzlich ein Git-Repository veröffentlicht [1], das erste Dokumente im OSCAL-Format bereitstellt. Diese Entwicklungen verdeutlichen, dass trotz der langjährigen Etablierung des IT-Grundschutz-Kompendiums die praktische Umsetzung und Verarbeitung

für viele Organisationen weiterhin eine Herausforderung darstellt.

Der im Rahmen dieser Arbeit entwickelte Prototyp bietet eine Grundlage, die in zukünftigen Arbeiten weiter ausgebaut werden kann. Insbesondere erscheint eine Integration der neuen OSCAL-basierten Bausteine sinnvoll, um die Informationsgewinnung sowie die Bewertung des Umsetzungsgrads weiter zu verbessern. Der prozessorientierte Ansatz des IT-Grundschutz++ stellt somit einen bedeutenden Fortschritt dar und kann als wichtiger Schritt in Richtung einer stärker automatisierten und standardisierten Sicherheitsbewertung betrachtet werden.

Abbildungsverzeichnis

Tabellenverzeichnis

Literaturverzeichnis

[1] Github repository bsi-bund. https://github.com/BSI-Bund/Stand-der-Technik-Bibliothek. [Online; Zugegriffen am 22.Januar.2026].

[2] AG, G. Greenbone ag - webseite. https://www.greenbone.net/blog/greenbone-security-feed-beinhaltet-mehr-als-100-000-schwachstellentests/, 2021. [Online; accessed 17-April-2025].

[3] AG, G. Greenbone ag - github. https://greenbone.github.io/docs/latest/background.html, 2025. [Online; accessed 17-April-2025].

[4] AG, G. Greenbone ag - webseite. https://www.greenbone.net/, 2025. [Online; accessed 17-April-2025].

[5] BSI. Umsetzungshinweise zum baustein app.3.1 webanwendungen. https://www.vultr.com/docs/how-to-install-openvas-vulnerability-scanner-on-ubuntu-16-04, 2016. [Online; accessed 04-August-2019].

[6] BSI. It-grundschutz-kompendium 1.edition 2018. https://www.bsi.bund.de, 2018. [Online; accessed 04-January-2019].

[7] BSI. Schwachstellen-analyse in netzen unter einsatz von openvas. https://www.allianz-fuer-cybersicherheit.de/ACS/DE/_/downloads/BSI-CS_007.pdf?__blob=publicationFile&v=5, 2018.

[8] BSI. Bundesamt für sicherheit in der informationstechnik - it-grundschutz-kompendium 2.edition. https://www.bsi.bund.de, 2019. [Online; accessed 10-February-2019].

[9] BSI. Bundesamt für sicherheit in der informationstechnik - it-grundschutz-kompendium edition 2020. https://www.bsi.bund.de, 2020. [Online; accessed 15-November-2020].

[10] BSI. Bundesamt für sicherheit in der informationstechnik - it-grundschutz-kompendium 2023. https://www.bsi.bund.de, 2023. [Online; accessed 10-April-2025].

[11] BSI. Aktuelle entwicklungen und ausblick zum it-grundschutz. https://www.bsi.bund.de/, 2025. [Online; accessed 05-June-2025].

[12] BSI. Bundesamt für sicherheit in der informationstechnik - webseite. https://www.bsi.bund.de, 2025. [Online; accessed 05-June-2025].

[13] BSI. Bundesamt für sicherheit in der informationstechnik - webseite. `https://www.bsi.bund.de`, 2025. [Online; accessed 17-April-2025].

[14] Colorlib. Adminlte. `https://adminlte.io/`, 2020. [Online; Zugegriffen am 20.Februar.2020].

[15] Dropwizard. Webseite. `https://www.dropwizard.io/en/stable/index.html`, 2016. [Online; Zugegriffen am 27.Jänner.2020].

[16] Dropwizard. Example. `https://github.com/dropwizard/dropwizard/tree/master/dropwizard-example`, 2020. [Online; Zugegriffen am 20.Februar.2020].

[17] Greenbone. Python gvm-api. `https://python-gvm.readthedocs.io/en/latest/`. [Online; Zugegriffen am 29.März.2019].

[18] Greenbone. 11.5.1 it-grundschutz. `https://docs.greenbone.net/GSM-Manual/gos-22.04/de/compliance-and-special-scans.html#it-grundschutz`, 2025. [Online; accessed 17-April-2025].

[19] Holm, H. Performance of automated network vulnerability scanning at remediating security issues. *Computers & Security 31*, 2 (2012), 164 – 175.

[20] Hu, Y., Sulek, D., Carella, A., Cox, J., Frame, A., and Cipriano, K. Employing miniaturized computers for distributed vulnerability assessment. In *2016 11th International Conference for Internet Technology and Secured Transactions (ICITST)* (Dec 2016), pp. 57–61.

[21] I. Yaqoob, S. A. Hussain, S. M. N. N. J. A., and u. Rehman, A. Penetration testing and vulnerability assessment. *Journal of Network Communications and Emerging Technologies (JNCET) 7*, 8 (8 2017).

[22] Intevation-GmbH. Openvas - webseite. `http://www.openvas.org/index.html`, 2018. [Online; accessed 20-December-2018].

[23] Intevation-GmbH. Openvas - tech doc portal. `https://docs.greenbone.net/API/GMP/gmp-7.0.html`, 2020. [Online; Zugegriffen am 20.Jänner.2020].

[24] Intevation-GmbH. Openvas - vulnerability-management. `https://docs.greenbone.net/GSM-Manual/gos-4/en/vulnerabilitymanagement.html`, 2020. [Online; Zugegriffen am 20.Jänner.2020].

[25] Karabasevic, D., Stanujkic, D., Brzaković, M., Maksimović, M., and Jevtić, M. Importance of vulnerability scanners for improving security and protection of the web servers. *Bizinfo Blace 9* (01 2018), 19–29.

[26] Mandal, N., and Jadhav, S. A survey on network security tools for open source. In *2016 IEEE International Conference on Current Trends in Advanced Computing (ICCTAC)* (March 2016), pp. 1–6.

[27] Reibold, H. *OpenVAS kompakt*. Bomots-Verlag, 2010.

[28] S. Fashoto, A., and Opeyemi, O. Evaluation of network and systems security using penetration testing in a simulation environment. *Computer Science and Telecommunications (GESJ) 54*, 2 (2018).

[29] Wang, Y., and Yang, J. Ethical hacking and network defense: Choose your best network vulnerability scanning tool. In *2017 31st International Conference on Advanced Information Networking and Applications Workshops (WAINA)* (March 2017), pp. 110–113.