Stage informatica AZ Turnhout

REALISATIEDOCUMENT

Jorik Goris
Student Bachelor in de Elektronica-ICT – Cloud & Cyber Security

2024 - 2025

Inhoudsopgave

1. Terminologie

Asset	Een waardevol onderdeel van de IT-omgeving, zoals een server, netwerkapparaat, applicatie of dataset.
Vulnerability	Een zwakke plek of fout in een systeem of applicatie die een aanvaller kan misbruiken om ongeoorloofd toegang te krijgen of schade te veroorzaken.
Threat	Een potentiële gebeurtenis of handeling (bv. malware, phishing, insider threat) die een asset negatief kan beïnvloeden.
CVE (Common Vulnerabilities and Exposures)	Een gestandaardiseerde catalogus van bekende kwetsbaarheden, elk met een uniek ID om wereldwijde uitwisseling van informatie te vergemakkelijken.
CVSS (Common Vulnerability Scoring System	Een schaal (0–10) om de ernst van kwetsbaarheden objectief te scoren op basis van impact, exploitbaarheid en omgevingsfactoren.
Pentesting	Een gecontroleerde, geautoriseerde aanvalssimulatie waarin experts proberen in een netwerk of applicatie in te breken om kwetsbaarheden te vinden.
Authenticated Scan	Een kwetsbaarheidsscan die met geldige credentials uitgevoerd wordt
Unauthenticated (Remote) scan	en scan zonder credentials, die open poorten onderzoekt alsof je een externe aanvaller bent.
XRD (Extended Detection and Response)	Een geïntegreerde security-oplossing die detectie en respons over meerdere bronnen (endpoint, netwerk, cloud, e-mail, applicaties) combineert.
EDR (Endpoint Detection and Reponse)	Technologie die verdachte activiteiten op endpoints (workstations, servers) monitort, analyseert en automatisch of handmatig reageert.
NAC (Network Access Control)	Mechanisme om netwerktoegang te regelen op basis van identiteit, compliance en locatie (bijv. wel/niet toestaan van devices).
VLAN (Virtual LAN)	Logische segmentatie binnen een fysiek LAN, zodat apparaten geïsoleerd kunnen communiceren zonder fysiek aparte netwerkapparatuur.
DMZ (Demilitarized Zone)	Een bufferzone tussen het interne netwerk en internet waar publieke servers (web, mail, VPN) staan, om interne systemen te beschermen.
SIEM (Security Information and Event Management)	Platform dat log- en eventdata centraliseert, correleert en analyseert om bedreigingen vroegtijdig te detecteren en forensisch onderzoek mogelijk te maken.
NIS2	U-richtlijn voor netwerk- en informatiebeveiliging, gericht op versterking van cyberweerbaarheid van essentiële en digitale dienstverleners.
CyFun (CyberFundamentals Framework)	Belgisch cybersecurity-raamwerk gebaseerd op NIST, ISO 27001 en CIS Controls, bedoeld als praktische invulling van risicogebaseerde beveiliging.

ESXi	De VMware bare-metal hypervisor-software waarmee je virtuele machines direct op fysieke servers draait.
vSphere	Het VMware-platform voor virtualisatie, met ESXi (hypervisor) én beheerlaag (vCenter) voor VM-lifecycle, clustering en resource-management.
vCenter	Centrale beheerconsole voor vSphere-omgevingen; biedt GUI/API voor provisioning, monitoring, updates en clustering van ESXi-hosts.
Type 1 Hypervisor	Hypervisor die rechtstreeks op de hardware draait (bare metal), zoals VMware ESXi, Microsoft Hyper-V of Proxmox.
Type 2 Hypervisor	Hypervisor die als applicatie op een bestaand besturingssysteem draait, zoals VMware Workstation of Oracle VirtualBox.
Docker	Containerplatform waarmee je applicaties en hun dependencies verpakt in geïsoleerde, lichtgewicht omgevingen.
Docker Compose	CLI-tool om multi-container Docker-applicaties te definiëren en beheren via een YAML-bestand.
Container	Geïsoleerde runtime-omgeving waarin een applicatie en al haar dependencies samen worden uitgevoerd, zonder volledige VM-overhead.
Kubernetes	Open-source orkestratieplatform voor automatisch uitrollen, schalen en beheren van container-gebaseerde applicaties.
RAID (Redundant Array of Independent Disks)	Technologie om meerdere fysieke schijven te combineren voor hogere beschikbaarheid (mirroring), performance (striping) of beide.
iDRAC (Integrated Dell Remote Access Controller)	Out-of-band managementmodule in Dell-servers voor remote BIOS-, firmware- en OS-beheer, onafhankelijk van het hoofdsysteem.
SSH (Secure Shell)	Beveiligd protocol voor command-line toegang tot (Linux/Unix-)servers, inclusief versleutelde bestandsoverdracht (SFTP, SCP).
RDP (Remote Desktop Protocol)	Microsoft-protocol voor grafische, externe desktoptoegang tot Windows-machines (en via xRDP ook Linux).
SFTP (SSH File Transfer Protocol)	Veilige file-transfer middels SSH, met encryptie en integriteit, vaak gebruikt voor geautomatiseerde data-uitwisseling.
Watchtower	Docker-container die periodiek draait en andere containers automatisch bijwerkt naar hun nieuwste image-versies.
Portainer	Gebruiksvriendelijke web-UI voor beheer van Docker- (en Kubernetes-)omgevingen, zonder CLI-afhankelijkheid.
DHCP (Dynamic Host Configuration Protocol)	Netwerkprotocol dat automatisch IP-adressen, subnetmaskers en andere configuratieparameters aan clients toewijst.
MAC-Spoofing	Handmatig wijzigen van het MAC-adres van een netwerkinterface om accesscontrolls of filters te omzeilen.
Advanced Hunting	Feature in Microsoft Defender waarmee security-analisten complexe queries uitvoeren op telemetry-data voor threat hunting.

Attack Surface Reduction (ASR)	Set regels in Microsoft Defender om risicovolle gedragspatronen (vb. ongewenste scripts, macros) preventief te blokkeren of te auditen.
SOC (Security Operations Center)	Team of facility dat 24/7 security-monitoring, detectie en incidentrespons levert.
ServiceNow (SNOW)	Enterprise-ITSM-platform voor incident-, probleem- en change-management, met workflow-automatisering en self-service portals.
Infoblox	Leverancier van geautomatiseerde DNS, DHCP en IPAM (DDI)-oplossingen voor centraal netwerk- en adresbeheer.

2. Inleiding

Het beheersen van kwetsbaarheden in een grote productieomgeving is geen eenvoudige opdracht, maar wel van cruciaal belang. Enerzijds wordt dit afgedwongen door regelgeving zoals de NIS2-wetgeving, anderzijds kan een slecht beveiligde IT-omgeving zware gevolgen hebben voor de continuïteit van productieprocessen of diensten binnen een bedrijf of organisatie.

Deze NIS2-richtlijn legt strengere eisen op aan organisaties die als kritisch worden beschouwd, zoals ziekenhuizen. Zij worden verplicht om maatregelen te nemen rond onder andere risicobeheer, incidentopvolging en het documenteren van kwetsbaarheden. Eén van de onderdelen uit deze richtlijn stelt dat organisaties actief hun kwetsbaarheden moeten identificeren en documenteren. Dat was meteen ook het uitgangspunt voor de hoofdopdracht van mijn stage.

Dit document beschrijft de technische uitwerking van een systeem dat ik tijdens mijn stage heb opgezet om kwetsbaarheden systematisch te detecteren en te documenteren. Om het project gestructureerd aan te pakken, werd in de eerste weken van mijn stage een plan van aanpak opgesteld. Daarin werd het probleem afgebakend en werden de stappen vastgelegd om tot een praktische oplossing te komen.

Hoewel deze hoofdopdracht centraal staat in dit document, kwamen er tijdens mijn stage ook andere relevante taken aan bod. Zo ontdekte ik tijdens het uitvoeren van de opdracht een verouderde SFTP-server, die jarenlang actief was en intussen end-of-life bleek. Ik onderzocht deze server en migreerde de volledige werking naar een nieuw, veilig platform. Daarnaast hielp ik mee bij het dagelijks opvolgen van beveiligingsincidenten via het Microsoft Defender-portaal. Hierbij behandelde ik onder andere phishingpogingen, analyseerde ik incidenten en werkte ik met de advanced hunting functie.

Het doel van dit document is niet alleen om de uitgewerkte oplossing toe te lichten, maar ook om te tonen hoe deze integreert in de bredere IT-omgeving van AZ Turnhout, en hoe deze bijdraagt aan het versterken van de algemene cyberveiligheid binnen de organisatie.

Bij deze wil ik ook uitdrukkend Hans Verboven, mijn stagementor bij AZ Turnhout, het gehele informatica-team, AZ Turnhout en de begeleidende docent, Liesbeth Kenens bedanken voor hun hulp en steun tijdens deze periode.

3. Voorstelling AZ Turnhout

AZ Turnhout (AZT) is een regionaal ziekenhuis dat bestaat uit twee campi. AZT is een fusie tussen het Sint-Elisabethziekenhuis en het vroegere Algemeen Ziekenhuis Sint-Jozef. In April van 2009 zijn beide ziekenhuizen gefusioneerd in het huidige AZ Turnhout. De officiële opstart vond plaats in September van dat jaar. Het nieuwe fusieziekenhuis kreeg de vorm van een vzw, waar verschillende partners deel van uit maken:

- Stad en OCMW Turnhout (voormalige uitbaters Sint-Elisabethziekenhuis)
- Vzw Kristelijke Medico-Sociale Instellingen (KMSI) (voormalige uitbater AZ Sint-Jozef)
- Vzw VZT (Vereniging Ziekenhuisartsen Turnhout)
- Vzw Huisartsen Vereniging Turnhout (HVRT) (Turnhout, kerncijfers, sd)

In dit nieuwe fusieziekenhuis krijgen artsen inspraak in het beheer ervan. Deze artsen zetelen in de algemene vergaderingen en hun experten in de raad van bestuur. Dit is een nieuw bestuursmodel dat vooruitstrevend is.

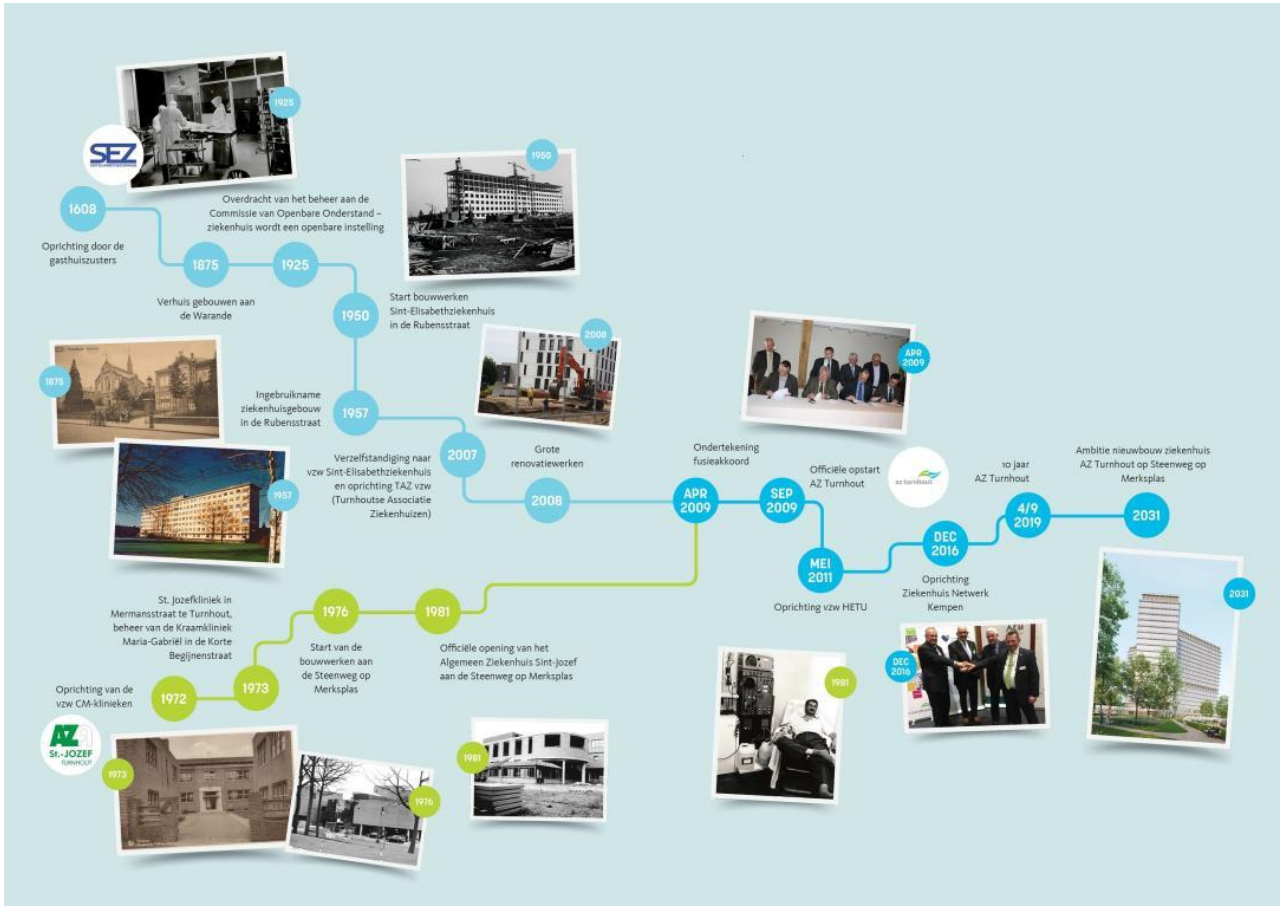
Ook is er een structurele samenwerking met de ziekenhuizen van Geel, Mol en Herentals onder de naam Ziekenhuisnetwerk Kempen (ZNK) wat ook op ICT-vlak invloedt heeft. (Turnhout, Ziekenhuisnetwerk Kempen, sd)

In het jaar 2031 heeft AZ Turnhout de ambitie om een nieuwbouw ziekenhuis te bouwen op Steenweg op Merksplas, waar nu de huidige campus Sint-Jozef bevindt. Dit nieuwe ziekenhuis zal de naam krijgen ‘Gasthuis in ’t groen’. (Turnhout, Gasthuis in 't groen, sd)

Gemiddeld telt AZT zo’n 22 500 opnames per jaar. AZT focust zich op specialisatie en een grote kwaliteit in patiëntenzorg. Verder werken er ruim 2200 mensen bij AZT. Dit zorgt ervoor dat AZ Turnhout bij de grotere werkgevers van Turnhout en omstreken hoort en dus ook en aanzienlijk grote ecomische invloed heeft op deze regio. Hieronder vindt u enkele cijfers van het jaar 2024.

- 2 160 medewerkers (+0%)
- 225 artsen (+0%)
- 105 vrijwilligers (+4%)
- 647 erkende bedden (0%)
- 442 686 consultaties (+2,5%)
- 49 964 spoedregistraties (-0,3%)
- 22 269 klassieke opnames (-6,7%)
- 14 092 ingrepen in dagziekenhuis (+2,7%)
- 1 514 bevallingen (-1,7%)
- 26 319 dialyses in onze 5 centra (+1,4%)

(% = evolutie ten opzichte van 2023) (Turnhout, AZ Turnhout in cijfers, 2023)



Figuur 3-1: Historiek AZ Turnhout, fusie Sint-Elisabethziekenhuis en Ziekenhuis Sint-Jozef

4. Analyse

4.1. Interne pentesting: waarom?

Een belangrijk punt voor de analyse is de opdracht zelf. Om een geschikte oplossing te kunnen ontwikkelen, is het essentieel om het probleem en de achterliggende vraag goed te begrijpen. De hoofdopdracht van mijn stage draaide rond het opzetten van een systeem dat interne pentesting toegankelijk maakt. De aanleiding hiervoor ligt in de NIS2-wetgeving. De NIS2-wetgeving is een richtlijn opgesteld door de Europese Unie.

Elk land of lidstaat is vrij om deze richtlijn om te zetten in wetgevingen, waaronder België. De Belgische wetgeving rond NIS2 is in werking gesteld op 18 oktober 2024. Vanaf deze datum hebben bedrijven 18 maanden de tijd om een Basic of Important verificatie te krijgen door een erkende en geaccrediteerde Conformity Assessment Body (CAB). Een CAB gaat de eigenlijke implementatie nakijken en hier een accreditatie voor toekennen.

Het Centre for Cybersecurity Belgium (CCB) omschrijft NIS2 als volgt:

“Het doel van de NIS2-wet is het versterken van maatregelen op het gebied van cyberveiligheid, incidentbeheer en toezicht voor entiteiten die diensten leveren die essentieel zijn voor het in stand houden van kritieke maatschappelijke of economische activiteiten. De wet heeft ook als doel de coördinatie van overheidsbeleid op het gebied van cyberveiligheid te verbeteren.” (CCB, sd)

Aangezien de gezondheidssector onder de zeer kritieke sectoren valt volgens de NIS2-richtlijnen, en AZ Turnhout meer dan 250 voltijdse equivalenten (VTE) telt, wordt het ziekenhuis beschouwd als een essentiële NIS2-entiteit. Dit betekent dat het moet voldoen aan zowel de ‘important’ als de ‘essential’ verplichtingen binnen de regelgeving om hiervoor gecertificeerd te kunnen worden.

Om de NIS2-wetgeving praktisch toepasbaar te maken, doen organisaties een beroep op een framework. Dergelijke frameworks helpen om de huidige beveiligingssituatie in kaart te brengen en om te bepalen waar verbeteringen nodig zijn. In België wordt vaak gebruik gemaakt van het CyberFundamentals Framework (CyFun), ontwikkeld door het CCB. Dit framework is gebaseerd op enkele andere veelgebruikte, internationale cybersecurity frameworks (NIST Cybersecurity Framework, ISO 27001/ ISO 27002, CIS Controls en IEC62443), maar afgestemd op de Belgische context. (nis2.be, sd)

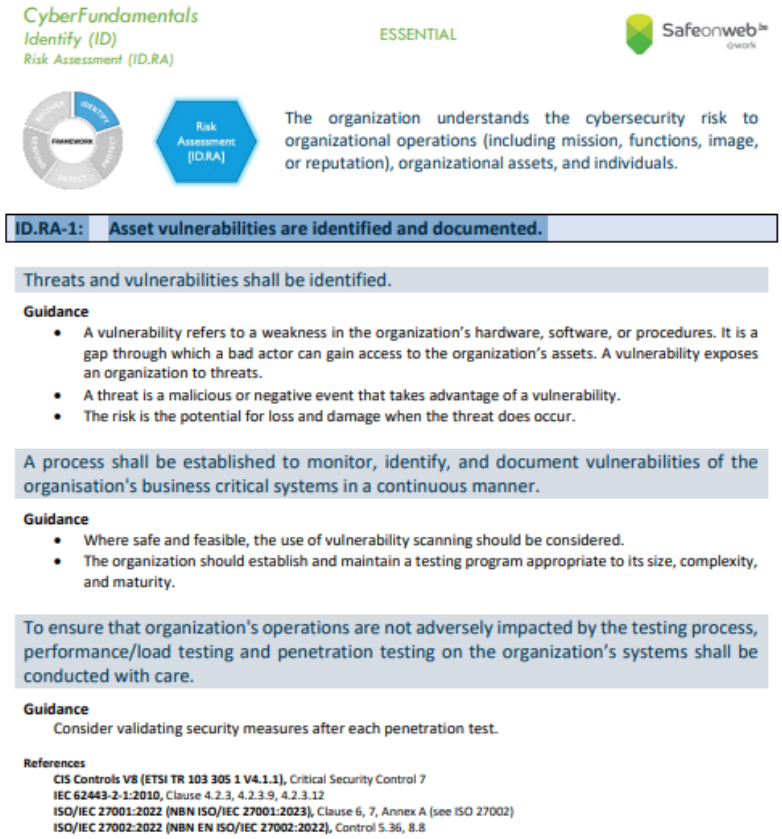
Elk EU-land heeft de vrijheid om eigen invulling te geven aan de concrete eisen van NIS2. Het gebruik van een framework is niet wettelijk verplicht, maar het vergemakkelijkt de implementatie aanzienlijk en zorgt voor een gestructureerde aanpak.

Bij AZ Turnhout maakt men momenteel gebruik van het CyFun-framework, al kan die keuze in de toekomst nog wijzigen. Binnen de Kempische ziekenhuizen wordt hierover regelmatig afgestemd en wordt er gekeken naar elkaars aanpak om tot een best practice te komen.

Het CyFun framework verdeelt zich in vijf kernfuncties; identify, protect, detect, respond en recover. Ieder van deze kernfuncties is onderverdeeld in categorieën. Een categorie onder de ‘identify’ kernfunctie kan bijvoorbeeld Asset Management (ID.AM) zijn. Zo kom je tot bijvoorbeeld ‘ID.AM-1 Inventarisatie van de binnen de organisatie gebruikte fysieke apparaten en systemen.’. Deze subcategorie bevat de specifieke eisen die het CyFun framework stelt rond Asset Management. (ccb, 2023)

Zo zijn er meerdere per subcategorie (ID.AM-1, ID.AM-2, ID.AM-3, ...). Onder de 'identify' kernfunctie kan dan ook weer een andere subcategorie vallen, bijvoorbeeld Business Environment (ID.BE). Zo krijg je bijvoorbeeld 'ID.BE-1: De rol van de organisatie in de toeleveringsketen wordt vastgesteld en gecommuniceerd.' Het CyFun framework is dus een goed gestructureerd framework. Doormiddel van de kernfuncties kan je je door het framework navigeren.

Onder de identify kernfunctie, met categorie risk assessment, vinden we de subcategorie "ID.RA-1: Asset vulnerabilities are identified and documented." die bestaat uit drie eisen:


CyberFundamentals
Identify (ID)
Risk Assessment (ID.RA)

ESSENTIAL

Safeonweb@work

Risk Assessment [ID.RA]

The organization understands the cybersecurity risk to organizational operations (including mission, functions, image, or reputation), organizational assets, and individuals.

ID.RA-1: Asset vulnerabilities are identified and documented.

Threats and vulnerabilities shall be identified.

Guidance

- A vulnerability refers to a weakness in the organization's hardware, software, or procedures. It is a gap through which a bad actor can gain access to the organization's assets. A vulnerability exposes an organization to threats.
- A threat is a malicious or negative event that takes advantage of a vulnerability.
- The risk is the potential for loss and damage when the threat does occur.

A process shall be established to monitor, identify, and document vulnerabilities of the organisation's business critical systems in a continuous manner.

Guidance

- Where safe and feasible, the use of vulnerability scanning should be considered.
- The organization should establish and maintain a testing program appropriate to its size, complexity, and maturity.

To ensure that organization's operations are not adversely impacted by the testing process, performance/load testing and penetration testing on the organization's systems shall be conducted with care.

Guidance

Consider validating security measures after each penetration test.

References

CIS Controls V8 (ETSI TR 103 305 1 V4.1.1), Critical Security Control 7
IEC 62443-2-1:2010, Clause 4.2.3, 4.2.3.9, 4.2.3.12
ISO/IEC 27001:2022 (NBN ISO/IEC 27001:2023), Clause 6, 7, Annex A (see ISO 27002)
ISO/IEC 27002:2022 (NBN EN ISO/IEC 27002:2022), Control 5.36, 8.8

Figuur 4-1: CyberFundamentals Framework richtlijn rond het identificeren en vastleggen van kwetsbaarheden

Deze eisen omschrijft mijn opdracht. De bedoeling is namelijk om:

- Threats (bedreigingen) en vulnerabilities (kwetsbaarheden) te gaan opsporen en identificeren.
- Een proces op te stellen om vulnerabilities op te sporen op kritische systemen.
- Pentesting met zorg uit te voeren, zodat de werking van de organisatie niet beïnvloed wordt door het testsysteem.

De laatste eis is een zeer belangrijk punt gezien de omgeving waarin deze opdracht werd uitgewerkt.

Aangezien we in een ziekenhuis omgeving zitten, werken we met een erg groot aantal aan kritische systemen (medisch, ondersteunend), die niet alleen invloed hebben op de werking van de infrastructuur en organisatie maar ook op mensenlevens en de kwaliteit van de zorg die we kunnen leveren.

In een ziekenhuisomgeving kom je in contact met veel verschillende systemen. Medische diensten maken gebruik van specifieke softwaretoepassingen, maar ook ondersteunende diensten zoals logistiek, de technische dienst, keuken en administratie hebben elk hun eigen toepassingen. Al deze software draait op achterliggende servers, die verschillende databanken en besturingssystemen (en versies) gebruiken.

Vergeet ook niet alle medische toestellen. Deze hebben meestal een volledige computer ingebouwd, of minstens een Operating System (OS). Deze zijn ook allemaal op het netwerk verbonden om met andere systemen te praten om bijvoorbeeld patiëntengegevens, data, beelden en werklijsten door te sturen. Ook deze apparaten kunnen kwetsbaarheden bevatten.

Tijdens mijn drie maanden stage bij AZ Turnhout kwam ik wekelijks in aanraking met één of meerdere nieuwe softwaretoepassingen, systemen of medische toestellen.

Dit alles maakt dat het opzetten van een pentestprocedure wel wat denkwerk vereist om ervoor te zorgen dat dit toepasbaar is in een grote, maar ook heel diverse productieomgeving. Volgende cijfers geven hier meer duiding bij.

AZ Turnhout in cijfers (Mei, 2025)

- 300+ (virtuele) servers
- 19 ESXi servers (verspreid over beide campi)
- 150 managed switches
- 3500+ endpoints (laptops, workstations, computers on wheels (cows), GSM's, tablets)
- 27 000+ identities

Microsoft Defender: huidige manier van vulnerabilitymanagement

Bij AZ Turnhout is Microsoft Defender uitgerold als Extended Detection & Response (XDR)-oplossing, in combinatie met Defender for Identity, Defender for Office 365 en Defender for Endpoint.

Microsoft Defender biedt een centraal portaal voor het opvolgen en beheren van securityincidenten. Daarnaast geeft het systeem aanbevelingen voor beveiligingsverbeteringen die toepasbaar zijn binnen onze IT-omgeving. Het is een zeer uitgebreide tool die ook inzicht geeft in vulnerabilities op apparaten die actief door Defender worden gemonitord.

Die monitoring geldt echter enkel voor toestellen die effectief onboarded zijn in Defender. Op dit moment zijn er ongeveer 3500 apparaten onboarded. Voornamelijk Windows-laptops, werkstations en servers worden beschermd door deze XDR-oplossing.

Een groot deel van de medische apparatuur valt hier echter buiten. Door beperkingen op het vlak van besturingssysteem, hardware, etc. kunnen deze niet worden geïntegreerd in Defender. Ook vallen veel IoT-systemen, camerasystemen, etc. hierbuiten. Toch bevatten ook deze systemen potentieel kwetsbaarheden, waarvan we graag op de hoogte willen blijven.

Om deze apparaten te kunnen detecteren en scannen op vulnerabilities, was er nood aan een aanvullende oplossing. Deze oplossing heb ik tijdens mijn stage uitgewerkt en opgezet.

Over Microsoft Defender wordt in het hoofdstuk rond mijn dagelijkse taken wat meer uitleg gegeven.

4.2. Scanning engine: Nessus vs. GVM (OpenVAS)

Om aan vulnerability scanning te doen hebben we een vulnerability scanner nodig. Dit zijn softwarepakketten die het mogelijk maken om met een paar simpele stappen een scan van een systeem of netwerk te starten. Deze scanners zijn op de hoogte van kwetsbaarheden door zogenoemde 'vulnerability feeds' en zullen deze gaan detecteren op een systeem.

Als deel van mijn analyse heb ik opzoekwerk gedaan rond enkele bekende opties om te gaan kijken welke een geschikte optie zou zijn om te gaan gebruiken.

De twee grote spelers in de (gratis) vulnerability scanning-markt zijn Nessus van Tenable en de Greenbone Vulnerability Manager (voorheen OpenVAS) van Greenbone.

Criteria	Nessus (CE)	GVM (OpenVAS)
Licentie	Nessus Essentials is gratis, bidet commerciële versie	Volledig open source (GPL)
Aantal CVE-checks	>50 000	+- 26 000
Detectie kritieke vulnerabilities	4,6% meer dan OpenVAS	Minder kritieke checks
Remote scans (unauth.)	Meer high & critical	Meer low & medium
Feed-updates	Automatisch	Manueel bij start
Interface & gebruiksgemak	Strakke, moderne interface	Minder intuïtief
Community & documentatie	Grote commerciële support, uitgebreide documentatie van Tenable	Actieve open source community, documentatie op Greenbone site

Er zijn zeker- en vast nog wat verschillen, maar mijn stage draaide niet rond het onderzoeken en vergelijken van vulnerability scanning tools. Bovenstaande criteria zijn enkele zaken die ik zelf tijdens het dagelijks gebruik heb opgemerkt.

Aangezien GVM gratis en open-source is, was het een logische keuze om deze te installeren op de scanning server. Nessus is een commercieel product, maar biedt ook een gratis Community Edition (CE) aan. Deze versie heeft iets minder functionaliteit dan de commerciële variant, maar blijft een zeer handige tool om in de toolbox te hebben.

Omdat beide scanners in een gratis versie beschikbaar zijn, heb ik ervoor gekozen om ze allebei te installeren en te documenteren. Daarnaast heb ik ook een vergelijking gemaakt tussen de twee, op basis van beschikbare informatie.

In het algemeen kan gesteld worden dat Nessus meer vulnerability checks aanbiedt dan OpenVAS volgens de vergelijking van Andy Hornegold (Hornegold, 2024) gaat het om ongeveer 10% meer. Daarnaast detecteert Nessus ook zo'n 5% meer 'critical' kwetsbaarheden (met een CVSS-score van ≥9).

Bij unauthenticated, remote scans ligt het iets genuanceerder. Een unauthenticated of remote scan is een scan waarbij er geen credentials van het doelwit worden meegegeven. De scanner zal, net zoals een aanvaller, vanop ‘afstand’ een scan uitvoeren op basis van welke poorten open staan, welke services aangeboden worden, welke versie van het besturingssysteem gebruikt wordt etc.

Een authenticated scan daarentegen krijgt SSH of Windows credentials van een echte, of testgebruiker mee om zo te kunnen ‘inloggen’ op het systeem en nauwkeuriger te kunnen gaan scannen op configuraties, instellingen, geïnstalleerde software etc.

OpenVAS scoort beter op het vlak van ‘low’ en ‘medium’ severity vulnerabilities, terwijl Nessus meer ‘high’ en ‘critical’ kwetsbaarheden detecteert in diezelfde context. Opvallend is ook dat OpenVAS vaak iets sneller is met het uitbrengen van remote checks voor nieuwe kwetsbaarheden. Meestal publiceren beide tools eerst een authenticated scan voor een bepaalde kwetsbaarheid, waarna de remote scan volgt.

4.3. Keuze van infrastructuur: Fysiek vs. virtueel

Voor het uitvoeren van vulnerability scanning was een dedicated server nodig. Hiervoor waren er twee opties: het opzetten van een virtuele machine (VM) binnen de bestaande ESXi-cluster, of het inzetten van een fysieke (bare-metal) server. Beide opties hebben voor- en nadelen.

De voorgenoemde ESXi-cluster, ook vaak vSphere & vCenter genoemd is een groepering van ESXi servers. ESXi is een OS (type 1 hypervisor) die virtualisatie voor VM's voorziet. Deze kunnen geclusterd worden om performantie- en management voordelen op te leveren. Binnen AZT is er een ESXi-cluster van 19 ESXi servers, verspreid over beide campi. ESXi is één server, vSphere is een product om deze te clusteren en vCenter is de toepassing waarop men dit systeem beheert. Deze termen worden soms echter los door elkaar gebruikt.

Voor- en nadelen fysieke server

Op vlak van security heeft een fysieke server de voorkeur. Aangezien we met een gesegmenteerd netwerk zitten, dat beveiligd wordt door Network Access Control (NAC), zitten we met een groot aantal Virtual Local Area Networks (VLANs).

Bij een gesegmenteerd netwerk gaan we een groter netwerk onderverdelen in kleinere segmenten. Dit heeft enkele voordelen. We kunnen het verkeer tussen deze segmenten gaan beveiligen doormiddel van een firewall, het broadcast verkeer gaan beperken wat voor betere prestaties zorgt en een efficiënter beheer gaan verwezenlijken.

Met VLANs maken we onze netwerksegmentatie mogelijk. We kunnen segmenteren door fysiek onze toestellen op andere apparatuur te scheiden, maar dit is in praktijk geen oplossing. Via VLANs kunnen we dit virtueel, zodat apparaten op dezelfde fysieke netwerkapparatuur toch logisch van elkaar gescheiden zijn.

Via NAC gaan we beperken welke toestellen toegang krijgen tot ons netwerk. We gaan op basis van MAC-adressen kijken of deze toestellen toegang mogen krijgen, en tot welk VLAN die toegang beperkt is. Met de hoeveelheid en aard van de toestellen in het netwerk is dit een systeem dat onmisbaar is in onze security portfolio.

Dit heeft als gevolg dat we bij een VM op vSphere niveau best wat netwerkwijzigingen zouden moeten doorvoeren. We zouden ervoor moeten zorgen dat alle VLANs die we willen scannen ook door de ESX worden aangeboden aan onze VM. Dit willen we voorkomen omdat we enkel de VLANs willen aanbieden die onze servers echt nodig hebben voor hun dagelijkse werking, wat vaak enkel het server-VLAN is. Hoewel verkeer tussen VLANs via de firewall verloopt, zou het plaatsen van de scanner in het server-VLAN extra firewallregels en configuratie vereisen, wat we liever niet doen. We willen geen regels instellen op de firewall die verkeer vanuit het server-VLAN naar alle andere VLANs toelaat, zo gaan we het nut van netwerksegmentatie tegen.

Bij een fysieke server kunnen we via NAC eenvoudig bepalen in welk VLAN deze geplaatst wordt. In AZ Turnhout maken we gebruik van Extreme Networks, waarbij we werken met één grote switching fabric. Dit laat toe om via NAC heel flexibel VLAN-toegang toe te wijzen aan fysieke toestellen. Deze switching fabric zorgt ervoor dat alle netwerkswitches binnen de omgeving met elkaar praten en als één grote switch gezien wordt.

Nadelen van een fysieke server zijn onder andere het hogere stroomverbruik, mogelijk ongebruikte resources (te veel werkgeheugen, rekenkracht), en het feit dat het onderhoud meer tijd vraagt. Ook is het minder eenvoudig om hier een backup van te nemen.

Voor- en nadelen virtuele server

Een virtuele machine (VM) als scanningsysteem biedt zeker enkele voordelen. Zo worden er automatisch back-ups gemaakt van de server en kunnen er snapshots (een soort backup van de huidige staat van de machine op een bepaald punt) genomen worden op momenten dat we configuraties aanpassen waarvan we de impact niet volledig kunnen inschatten.

Daarnaast verbruikt een VM veel minder resources, zowel op vlak van stroom als rekenkracht, in vergelijking met een fysieke server. Ook het herinstalleren van het besturingssysteem is eenvoudiger bij een VM, aangezien dit volledig op afstand kan gebeuren.

Op de fysieke server die ik voor deze opdracht heb gebruikt, heb ik wel de Integrated Dell Remote Access Controller (iDRAC) geconfigureerd, wat op afstand beheren en herinstalleren ook mogelijk maakt. Toch is dit minder gebruiksvriendelijk dan bijvoorbeeld het redeployen van een VM via VMware vCenter met een vooraf ingestelde template. Dit is momenteel het grootste verbeterpunt van mijn oplossing.

Gemaakte keuze en ervaring

TABEL KEUZE FYSIEK OF VM INVOEREN

Uiteindelijk heb ik ervoor gekozen om de opdracht uit te werken op een fysieke server, namelijk een Dell R640 waarop eerder tijdelijk een network monitoring tool draaide. Het is nog een vrij recente en performante server die beschikbaar was.

De belangrijkste reden voor deze keuze was dat het een stuk handiger is om via NAC te werken met een fysieke server. Daarnaast komt er bij deze aanpak ook wat datacenterwerk kijken, zoals het installeren, monteren en onderhouden van de server, wat ook nuttige praktijkervaring oplevert.

Achteraf bekeken ben ik tevreden met deze keuze, al is wel gebleken dat het werken met een fysieke server op afstand, zeker bij netwerkaanpassingen, niet zonder risico is.

De server draait namelijk in het datalokaal op de campus Sint-Jozef, terwijl ik zelf dagelijks werk op de campus Sint-Elisabeth. Het is wel eens voorgekomen dat het aanpassen van netwerkconfiguraties ervoor zorgde dat ik de SSH- of RDP-verbinding verloor, waardoor ik fysiek naar de andere campus moest om dit lokaal op te lossen. Op een virtuele machine had ik dat soort problemen gewoon vanop afstand kunnen oplossen.

SSH en RDP zijn manieren om vanop afstand met een server of systeem te verbinden. Via SSH kan je commando’s naar een server sturen, terwijl RDP een volledige desktopervaring zal aanbieden.

4.4. Spoofing: Scripted via NAC vs. dedicated test VLAN

Deze analyse bepaalt de manier waarop de onze server van VLAN gaan verplaatsen.

Hiervoor waren er twee mogelijke opties. De eerste optie was om het MAC-adres van één van de netwerkinterfaces te spoofen, oftewel het originele adres aanpassen, zodat de NAC denkt dat het om een bekend apparaat gaat. Nadien zal de NAC in zijn configuratie kijken en de server in een vooraf geconfigureerd VLAN plaatsen. Nadien zal het toestel via DHCP een IP-adres toegewezen krijgen in dat VLAN. DHCP is een netwerkservice die onderandere IP-adressen aan een toestel kan toewijzen.

De tweede optie was het aanmaken van een dedicated test-VLAN, waarin de server permanent zou blijven. Vervolgens zou via firewallregels de routering tussen dit test-VLAN en de te scannen VLANs worden ingesteld. Hierbij wordt gekeken naar de bron van de pakketten, waar ze naartoe moeten en wat hiervoor de beste weg is. De best weg is niet altijd de snelste, aangezien deze soms pakketverlies kan hebben.

Die tweede optie werd al snel uitgesloten, aangezien dit veel configuratiewerk op de firewall vereist en bovendien een aanzienlijk beveiligingsrisico met zich meebrengt. Als er op een of andere manier een ander toestel toegang krijgt tot het test-VLAN, zou dit toestel in principe ook toegang kunnen krijgen tot alle andere VLANs, tenzij je op de firewall de source van de verbindingen strikt beperkt tot het IP-adres van de scanning server, wat de configuratie nog complexer maakt.

Omdat NAC al in gebruik was, en omdat het spoofen van het MAC-adres vrij eenvoudig kon met een handig Python-scriptje, is uiteindelijk voor die aanpak gekozen.

4.5. Migratie SvDockerHost01: upgrade type analyse

Deze analyse komt voort uit een van de resultaten van de interne pentesting, een outdated, kwetsbare Ubuntu server die als SFTP-server functioneert binnen de DMZ. De DMZ, of ‘Demilitarized Zone’, is een buffernetwerk tussen het interne netwerk en het internet. Hier worden servers geplaatst die toegankelijk moeten zijn vanop het internet, maar we geen toegang willen geven tot ons eigen netwerk. Aangezien er in het huidige infra-team geen kennis van Docker is, heb ik dit project erbij genomen tijdens mijn stage.

Voor de migratie van SvDockerHost01, de kwetsbare server, naar een veiligere, geüpdatete versie waren er een aantal mogelijke opties. De server dateert van 2018 en draait nog op Ubuntu 18.04 LTS, waarvan de ondersteuning afliep op 31 mei 2023. Bovendien is deze server rechtstreeks bereikbaar vanaf het internet. Op deze host draaien vijf SFTP-servers in Docker-containers, gebaseerd op Ubuntu 16.04 (end-of-life sinds 2021). Ook de Docker versie zelf was al een 8-tal versies verouderd.

Tijdens het onderzoek van de server, waar geen enkele documentatie over bestond en waar het huidige infra-team geen ervaring mee had, bleek uit de logs dat er veel brute-force aanvallen vanuit het internet plaatsvonden.

Bij het uitscannen van deze server via de pentest oplossing (de hoofdopdracht), bleek ook dat hier een groot aantal kritieke kwetsbaarheden in aanwezig zijn. Dat wil je hoe dan ook vermijden wanneer een server ook nog eens vanop het internet bereikbaar is. Wel was deze gelukkig goed gesegmenteerd in een apart FTP-DMZ VLAN.

Mogelijke upgrade opties

De volgende scenario’s werden overwogen:

- Een back-up maken van de huidige server en een in-place upgrade uitvoeren
- Een nieuwe server van scratch opzetten en alles migreren
- Overschakelen naar een alternatief systeem zoals Talos Linux
- De containeroplossing overzetten naar een Kubernetes (K8s) cluster

De eerste optie werd al snel uitgesloten. De kloof tussen Ubuntu 18.04 en de huidige LTS-versie (24.04) is simpelweg te groot, een verschil van 6 jaar. Zo'n grote sprong in versies verhoogt de kans op foutmeldingen, dependency-issues en incompatibiliteiten aanzienlijk. Een in-place upgrade van een Linux-server is bovendien minder betrouwbaar en complexer dan bij Windows: package-versies kunnen conflicteren, en foutherstel is moeilijker.

Ook de opties Talos Linux of een volledig Kubernetes-cluster zijn even overwogen. Hoewel ik hier ervaring mee heb, was dit voor deze context overkill. De huidige SFTP-servers, hoewel licht misgeconfigureerd, draaiden al zeven jaar stabiel zonder grote incidenten. Het risico bestond dat het invoeren van een complexere oplossing meer problemen zou veroorzaken dan het simpelweg moderniseren van de bestaande configuratie.

Een SFTP-server is een FTP-server, die gebruik maakt van de SSH-daemon om authenticatie en encryptie te voorzien. Via deze servers kunnen bestanden uitgewisseld worden tussen verschillende systemen op een veilige manier. Deze worden het ziekenhuis bijvoorbeeld gebruikt voor communicatie tussen het Rode Kruis (voor bloedvoorraden), en het labo. In dit geval bestaan deze servers uit docker-containers.

Een docker-container is een kleine, lightweight en geïsoleerde omgeving die alles bevat om een applicatie, bijvoorbeeld FTP of een webserver te kunnen draaien. Deze containers zijn platformonafhankelijk en bieden ontwikkelaars een mogelijkheid om applicaties te ontwikkelen die op elke infrastructuur (waar docker op geïnstalleerd is) kunnen draaien.

Gekozen aanpak

Ik heb gekozen voor een volledig nieuwe, geüpdatete server (24.04) met een verse Docker-installatie. Deze keer heb ik gebruik gemaakt van Docker Compose, wat in de vorige setup ontbrak. Met docker-compose kunnen we makkelijk meerdere docker-containers gaan configureren die invloed hebben op elkaar. In ons geval vijf-SFTP servers. Dit biedt ons de mogelijkheid erg makkelijk en overzichtelijk de configuraties te gaan koppelen, of om de servers te herstarten. Docker-compose wordt vaak gebruikt wanneer applicaties meerdere containers nodig hebben. Denk aan een webserver, database en API-container.

Deze oplossing maakt het beheer een stuk overzichtelijker. Containers starten automatisch opnieuw op bij crashes of herstarts van de server, iets wat in de vorige situatie niet voorzien was.

In de oude setup werden de containers handmatig gestart via docker run-commando's. Daarbij moesten configuratiemappen handmatig worden gekoppeld, en werd de configuratie vaak achteraf aangepast binnenin de container zelf, door extra mappen aan te maken, rechten aan te passen, enzovoort.

Bij een herstart of crash van de server leidde dit tot SFTP-containers zonder correcte configuratie. Door de overstap naar Docker Compose is alle configuratie nu centraal vastgelegd in één YAML-bestand, waardoor herstarten probleemloos verloopt en wijzigingen eenvoudiger doorgevoerd kunnen worden.

Daarnaast heb ik ook Portainer geïnstalleerd, zodat het infra-team in de toekomst eenvoudiger met de containeromgeving aan de slag kan. Via deze gebruiksvriendelijke webinterface kunnen ze, zonder diepgaande Docker-kennis:

- Containerlogs bekijken
- Containers openen via een ingebouwde console
- Statistieken raadplegen
- Basisbeheer uitvoeren (starten, stoppen, herstarten)

De nieuwe oplossing is niet alleen veiliger en stabieler, maar ook veel makkelijker beheersbaar voor het team. De nieuwe server biedt voordelen op verschillende vlakken:

- Veiligheid van de host server
- Veiligheid van de container images

- Gebruik van Docker Compose voor beheer en herstartbaarheid
- Betere documentatie van de omgeving
- Gebruik van Portainer voor visueel en toegankelijk beheer

5. Realisaties

Dit hoofdstuk en de volgende bevatten de beschrijving van het resultaat dat je hebt gemaakt: de functionaliteiten, de opzet van de toepassing, bepaalde principes die je hebt toegepast, welke patronen je geïmplementeerd hebt…
Deze beschrijving moet niet volledig zijn maar moet een goed idee geven van wat je hebt gemaakt. Kleine stukjes code ter illustratie, bepaalde schermen en hoe die werken, enz.… staan hier op hun plaats.

5.1. Internal pentesting: NIS2 compliance: hoofdopracht

Dit hoofdstuk beschrijft de realisatie van mijn hoofdopdracht, namelijk het opzetten van een systeem om te voldoen aan de “ID.RA-1: Asset vulnerabilities are identified and documented.” vereiste van het CyFun-framework.

5.1.1. Project scope

Allereerst is het belangrijk om de scope van het project te beschrijven. De oorzaak van het probleem is dat onze huidige XDR-oplossing, Microsoft Defender, niet in alle hoekjes van het netwerk zicht heeft. De toestellen die vooral buiten deze oplossing vallen zijn medische toestellen, niet-Windows toestellen en IoT-systemen.

NIS2 verwacht van ons dat we van alle kritische systemen zicht hebben op de vulnerabilties die hier aanwezig zijn, en een manier hebben om deze te documenteren. Dit valt onder de grotere term ‘vulnerability management’, of het beheer van kwetsbaarheden. Om ze te kunnen oplossen, en ervoor te zorgen dat de kritische systemen van de organisatie beschikbaar blijven, moeten we dus een vorm van vulnerability scanning hanteren.

Binnen de scope van het project valt het opstellen van een systeem om vulnerability scanning in een groot netwerk efficiënt op te stellen, dit systeem te testen, te documenteren en over te dragen op een manier dat dit in de toekomst met een bepaalde frequentie kan herhaald worden.

Het exploiteren van vulnerabilties valt buiten de scope van dit project.

5.1.2. Technische implementatie

Zoals in het analyseonderdeel van dit document beschreven, heb ik de keuze gemaakt om een fysieke server te gebruiken voor de installatie. Dit is een server die zich in het datalokaal van de campus Sint-jozef bevindt, en eerder gebruikt werd voor een networkmonitoring tool. Deze werd echter niet meer gebruikt, en was dus de perfecte kandidaat om onze scanning server te worden. De andere opties waren om een oude laptop of desktop te gebruiken, maar dit had ook wat nadelen zoals de beperkte rekenkracht en beperking qua netwerk interfaces.

Aangezien we voor de uitwerking voor een fysieke server hebben gekozen, geeft de aanwezigheid van meerdere netwerkkaarten ons wat voordelen. Dit wordt verduidelijkt in het netwerkschema.

Hieronder een overzicht met details over de server die gebruikt werd.

Specificatie	Waarde
Merk	Dell
Type	R640
RAM	64GB DDR-4 (2666 Mhz)
CPU 1	Intel(R) Xeon(R) Gold 6128 PCU @ 3.40Ghz (6C)
CPU 2	N/A
Integrated NIC 1	Intel(R) 2P X520/2P I350 rNDC
NIC Slot 1	BRCM GbE 4P 5719-t Adapter
Solid State Disk 0:1:0	240GB
Solid State Disk 0:1:1	2000GB
Solid State Disk 0:1:2	2000GB
Raid configuratie	Raid 5
OS	Kali Linux 2025.1
hostname	svkali
Management IP	10.131.130.96

Tabel 5-1: Specificaties server die gebruikt werd voor het uitvoeren van vulnerabilityscanning

Zoals voorheen aangehaald kunnen we bij een virtuele server makkelijk backups maken van de server. Bij een fysieke server is dit niet het geval. Daarom is het belangrijk dat we de opslag, waarop ons besturingsysteem draait, redundant maken.

Om dit in praktijk te verwezenlijken maken we gebruik van een Redundant Array of Independent Disks (RAID), waarbij we meerdere schijven gaan bundelen als 1 logische schijf of volume. Op dit volume, dat onderliggend bestaat uit meerdere harde schijven kunnen we ons besturingssysteem installeren.

Een opmerking die ik hierbij wil maken is dat een RAID 5 op een array met disks van een verschillende grootte niet optimaal is.

Op deze manier houden we slechts 446GB over van de 4.5TB die we oorspronkelijk hadden. Wanneer we bijvoorbeeld drie disks van 2000GB zouden gebruiken, houden we 3.84TB over met 1.92TB aan redundantie.

Er was ook de optie om voor een RAID 1 te kiezen, een zogeheten mirror van de twee 2000GB disks. Dit zou ons meer opslag geven, maar iets minder performantie. In praktijk zou dit waarschijnlijk een even goede oplossing geweest zijn.

Echter waren de huidige disks nog aanwezig in de server en waren ze vrij nieuw. Ook zal er niet veel data op deze server opgeslagen worden, en is het verlies van die capaciteit dus acceptabel.

Gezien er een besturingssysteem op geïnstalleerd staat is het interessanter om te kiezen voor een RAID 5 die snellere lees- en schrijf performantie aanbiedt dan een RAID 1.

Virtuele disks (iDRAC):

	Status	Name	State	Layout	Size	Media Type	Read Policy	Write Policy	Stripe Size	Secured	Remaining Redundancy
+	☑	vdk0	Online	RAID-5	446 GB	SSD	Read Ahead	Write Back	64K	No	1

Figuur 5-1: virtual disk in iDRAC als resultaat van RAID configuratie

Fysieke disks (iDRAC):

	Status	Name	State	Slot Number	Size	Security Status	Bus Protocol	Media Type	Hot Spare	Remaining Rated Write Endurance
+	☑	Solid State Disk 0:1:0	Online	0	223 GB	Not Capable	SATA	SSD	No	100%
+	☑	Solid State Disk 0:1:1	Online	1	1787.88 GB	Not Capable	SATA	SSD	No	100%
+	☑	Solid State Disk 0:1:2	Online	2	1787.88 GB	Not Capable	SATA	SSD	No	100%

Figuur 5-2: fysieke disks in iDRAC

Installatie server & scanning tools

Voor het besturingsysteem heb ik gekozen voor Kali Linux. Dit besturingssysteem komt standaard met een groot aantal aan handige tools die gebruikt kunnen worden tijdens pentesting.

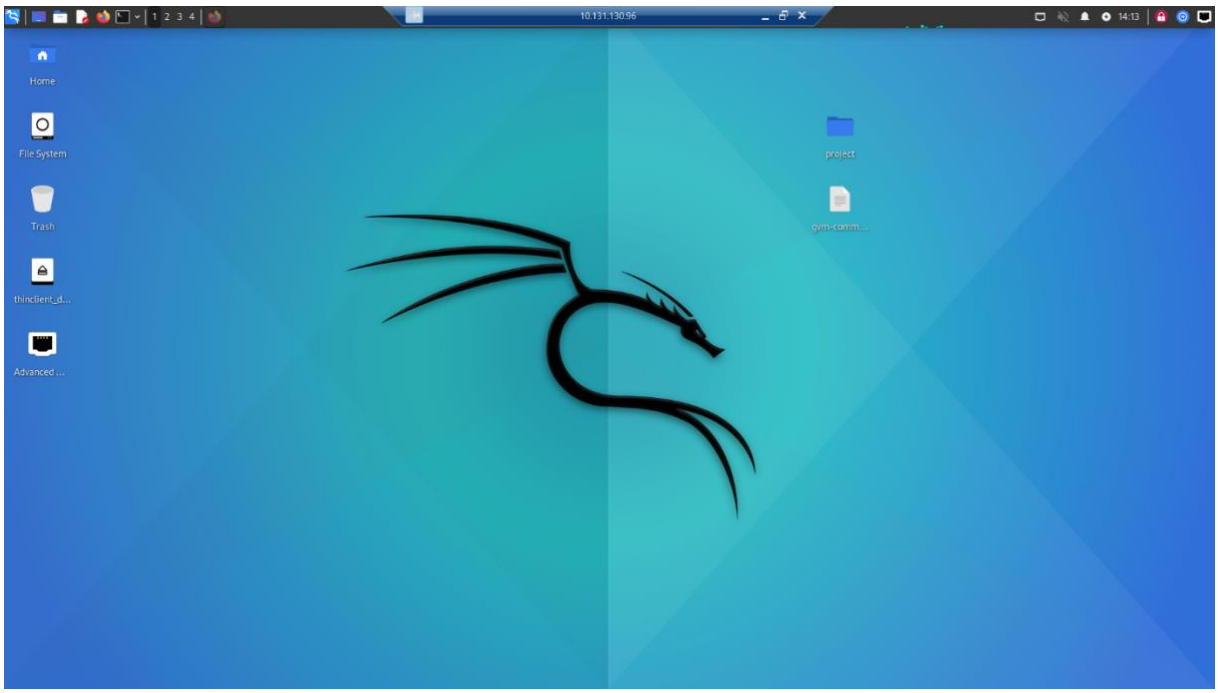
De installatie van de server verliep vrij eenvoudig. Ik heb de Kali Linux .ISO gedownload en hiermee een bootable USB-stick aangemaakt. Die heb ik vervolgens gebruikt om het besturingssysteem te installeren op de RAID 5-configuratie van de server. Dit proces verliep zonder problemen.

Wel had ik tijdens de eerste installatie één ding over het hoofd gezien: in de nieuwe Kali-installer staat de installatie van een desktopomgeving (GUI) standaard uitgeschakeld. Hierdoor had ik na de installatie alleen een command-line interface (CLI) beschikbaar.

Voor een serverversie van Ubuntu zou dat geen probleem zijn, maar bij Kali Linux is een GUI toch wel handig, zeker voor het gebruik van tools zoals Nessus of OpenVAS met een grafische interface.

Na wat opzoekwerk heb ik geprobeerd om de desktop achteraf te installeren, maar uiteindelijk heb ik ervoor gekozen om de server volledig opnieuw te installeren, deze keer mét GUI.

Na de installatie heb ik ook nog de powersettings aangepast, zodat de server niet automatisch in slaapstand gaat. Aangezien deze server op afstand beheerd wordt, zou het uiteraard niet handig zijn als hij zichzelf uitschakelt of in slaap valt wanneer hij niet gebruikt wordt. Na deze stappen had ik een werkende Kali Linux.



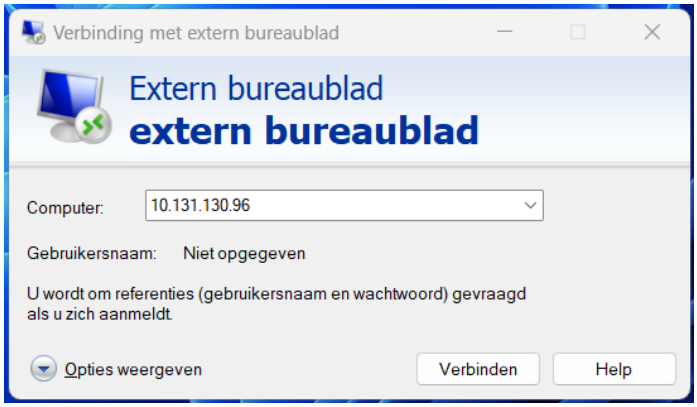
Figuur 5-3: Desktop Kali Linux na installatie

Nu is het tijd om ervoor te zorgen dat we deze ook vanop afstand kunnen beheren. Deze fysieke machine draait namelijk zoals reeds vermeld in het datalokaal van campus Sint-Jozef, terwijl de dienst informatica zich in campus Sint-Elisabeth bevindt. Het zou erg onhandig zijn moest iemand fysiek naar dit datalokaal gaan om daar een scherm, toetsenbord en muis aan te sluiten om zo te kunnen gaan scannen.

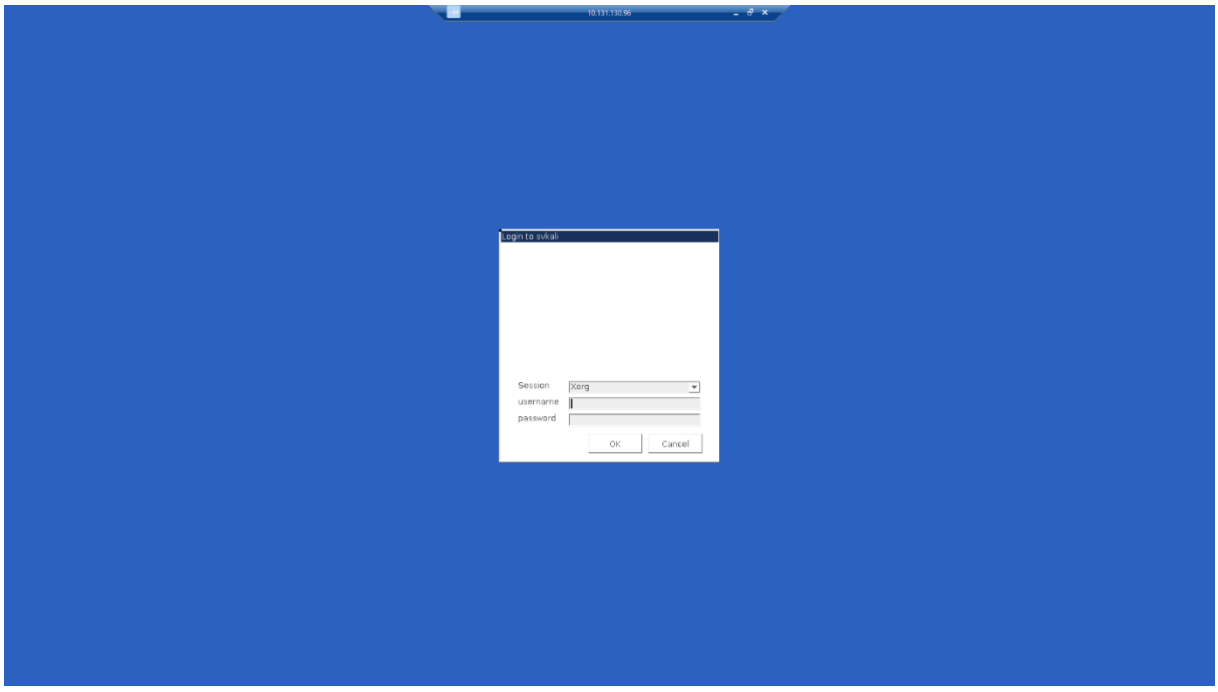
Om een computer vanop afstand te beheren wordt dikwijls gebruik gemaakt van het Remote Desktop Protocol (RDP). Op Linux kunnen we xRDP gebruiken, een alternatief voor de Windows versie.

XRDP:

```
$ sudo apt get update
$ sudo apt install xrdp –y
$ sudo apt install xorgxrdp
$ update-rc.d xrdp enable
$ systemctl xrdp start
$ systemctl xrdp enable
```



Figuur 5-4: RDP wizard voor het verbinden met Kali Linux over xRDP

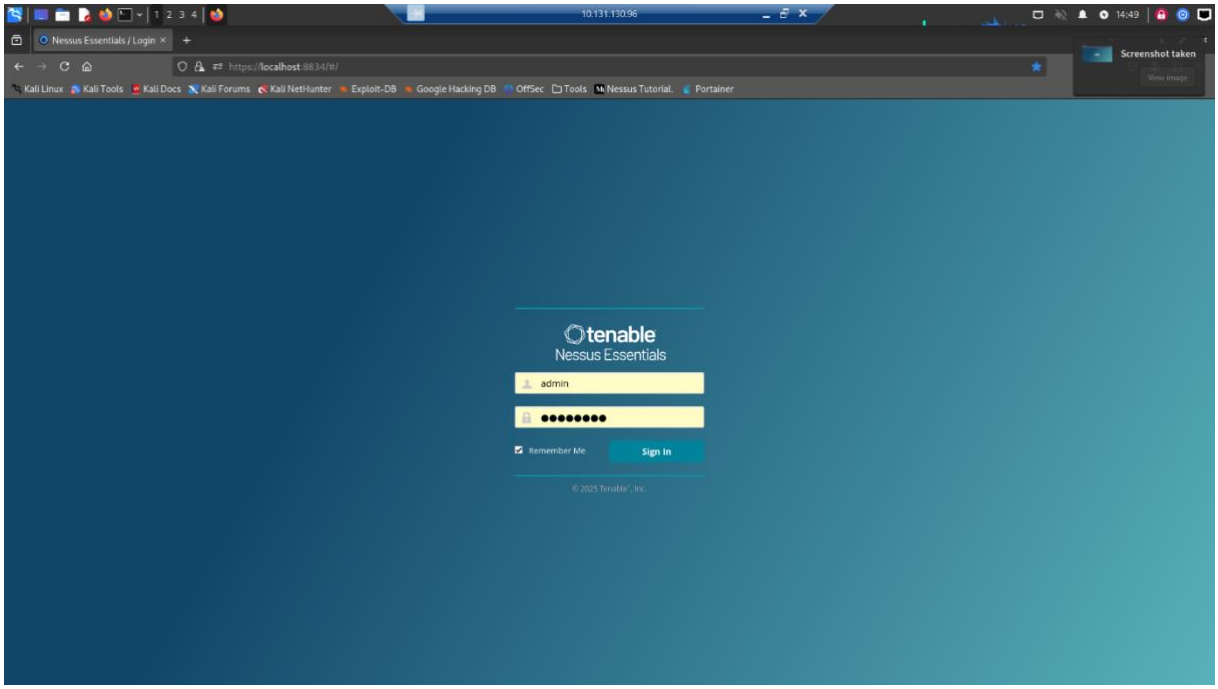


Figuur 5-5: Authenticatie scherm van xRDP

Nu kunnen we via RDP verbinden op onze server, en deze vanop afstand beheren. Er zijn nog twee zaken die geïnstalleerd moeten worden, namelijk de scanning tools zelf. In dit geval is dat Nessus en OpenVAS.

Nessus:

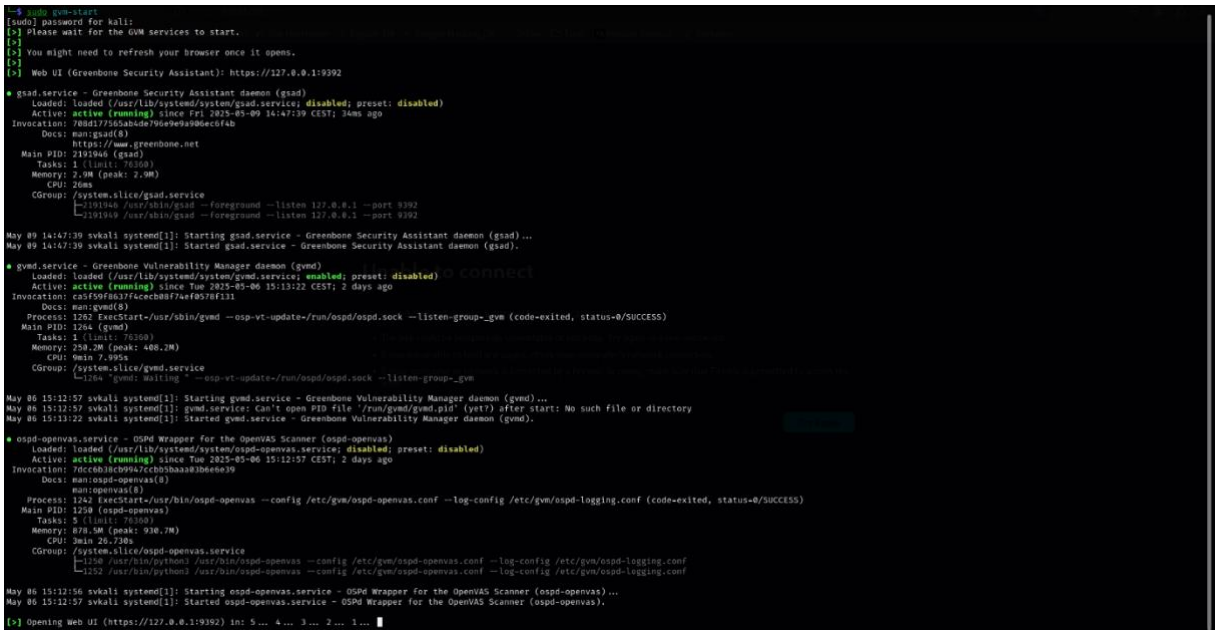
```
$ curl --request GET \ --url
'https://www.tenable.com/downloads/api/v2/pages/nessus/files/Nessus-10.8.4-
debian10_amd64.deb' \ --output 'Nessus-10.8.4-debian10_amd64.deb'
$ dpkg –i Nessus-10.8.4-debian10_amd64.deb
$ systemctl start nessusd
$ systemctl enable nessusd
```



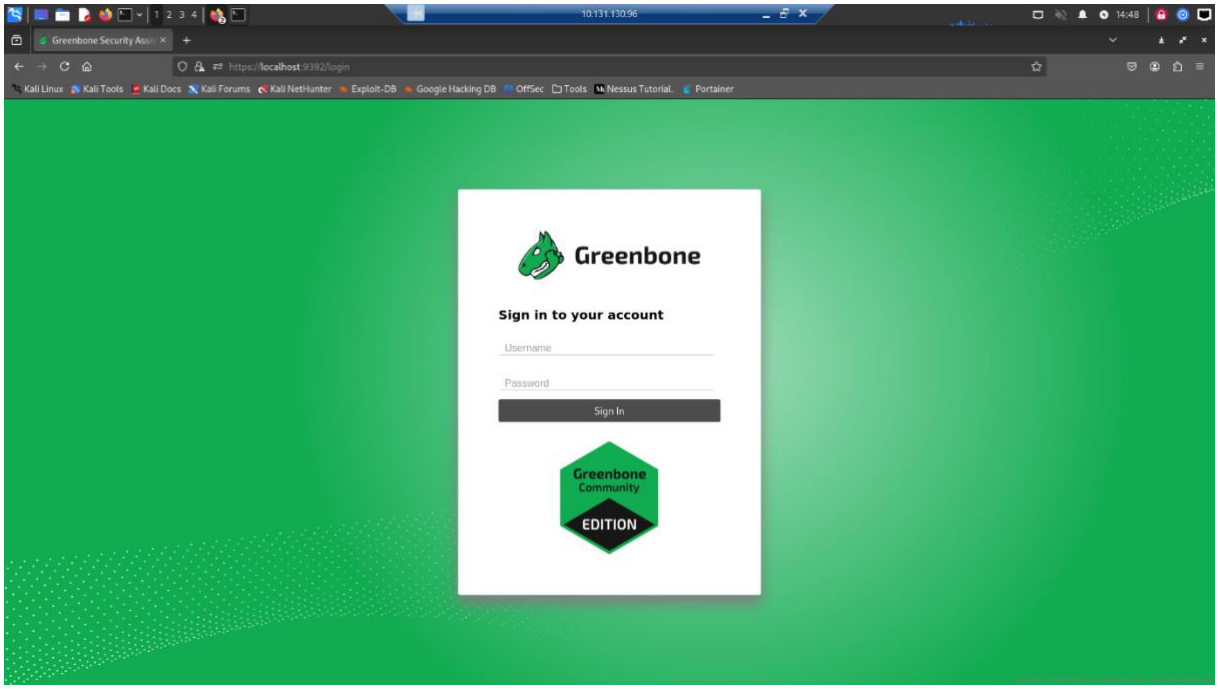
Figuur 5-6: Login scherm Tenable Nessus, na sucessvolle installatie

OpenVAS:

```
$ sudo apt update && sudo apt upgrade –y
$ sudo apt install openvas –y
$  sudo gvm-setup
(Log het paswoord)
$ sudo gvm-check-setup
```



Figuur 5-7: Starten van Greenbone Vulnerability Manager via commandline



Figuur 5-8: Login scherm Greenbone Vulnerability Manager na sucessvolle installatie

Persoonlijk heb ik ondervonden dat de installatie van Nessus een stukje makkelijker ging. Het downloaden en installeren van een .deb file is vele malen sneller dan de packages downloaden en installeren via apt.

Dit kan je vergelijken met het downloaden van een .exe bestand op Windows en deze dan uitvoeren om alles te installeren. Alles dat je nodig hebt om het programma te doen werken zit in dit bestand. Wanneer je iets via apt gaat installeren, installeer je de losse componenten die hiervoor nodig zijn. Het kan dus zijn dat er soms een versieconflict ontstaat. Dit is moeilijker op te lossen.

Ook heb ik tijdens het tijdens het installeren van OpenVAS enkele problemen ondervonden.

Na het uitvoeren van het ‘sudo gvm-start' commando om OpenVAS te initialiseren en installeren gaat OpenVAS ook de ‘vulnerability feeds’ downloaden. Deze vulnerability feeds zijn een soort lijst of verzameling van alle huidige kwetsbaarheden. Deze worden aangevuld door informatie van producenten van hardware of software, of door security-onderzoekers.

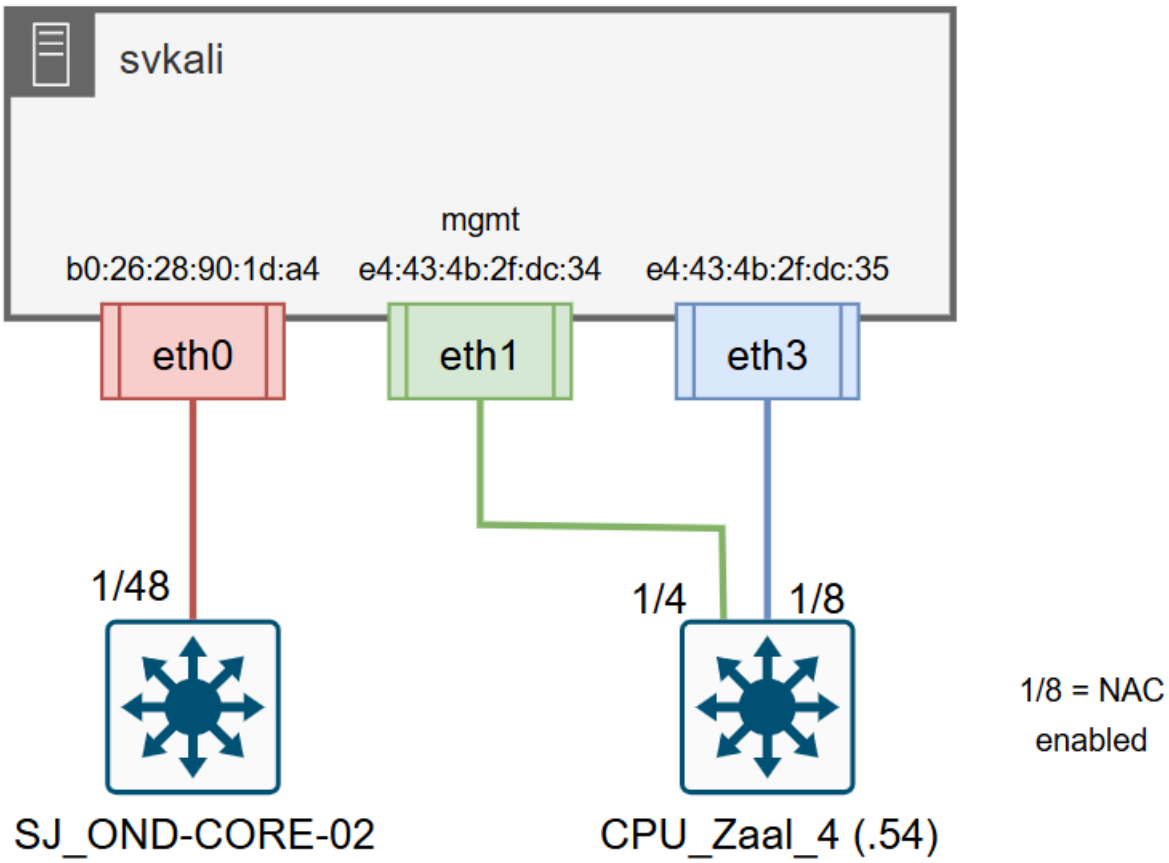
Dit proces crashte enkele keren, waarna ik tot de conclusie kwam dat onze firewall enkele van deze pakketten tegenhoudt en de verbinding met de servers waar deze threat intelligence (informatie rond kwetsbaarheden) van gedownload wordt geblokkeerd werd.

Als tijdelijke oplossing heb ik de server gepatched op de ‘hotspot’ lijn, een soort verbinding die op enkele poorten aanwezig is in de datalokalen en de informaticadienst die niet langs de firewall gaat, maar apart langs een ordinaire Telenet modem gaat. Dit kan handig zijn om firewall problemen te troubleshooten.

Via de hotspot lijn lukte het installeren en downloaden van de vulnerability feeds wel, waarna ik OpenVAS kon starten.

Schematisch overzicht & netwerk

Om de opstelling van de server binnen het netwerk te verduidelijken heb ik volgend schema gemaakt.



Figuur 5-9: Netwerkschema server

Zoals je kan zien, is svkali verbonden met twee verschillende switches, en dat is met een goede reden.

CPU_ZAAL_4 is een zogenaamde edge switch, wat betekent dat hier voornamelijk clients op aangesloten zijn. Op dit type switch zijn de poorten vaak NAC-enabled, omdat we willen controleren welke apparaten toegang krijgen tot het netwerk, en tot welk VLAN ze vervolgens behoren. Deze poort gebruiken we als onze dynamische ‘scanpoort’, omdat we hier met de NAC-configuratie kunnen spelen om te bepalen in welk VLAN de poort terechtkomt en dus welke VLAN we gaan scannen.

SJ_OND-CORE-02 is daarentegen een core switch. Je kan dit zien als de backbone van het netwerk. Hierop zijn geen eindgebruikers aangesloten, maar vooral andere switches en de ESX-servers. Op deze core switch zijn alle VLANs aanwezig en hoeven ze dus niet zoals bij edge switches “doorgevoerd” of “allowed” te worden.

Wanneer we een VLAN willen scannen dat niet via NAC wordt beheerd, of geen DHCP heeft, moeten we dit doen met een statisch IP-adres op onze netwerkinterface (bijvoorbeeld eth0). Om dan alsnog toegang te krijgen tot het juiste VLAN, moeten we gebruikmaken van VLAN-tagging.

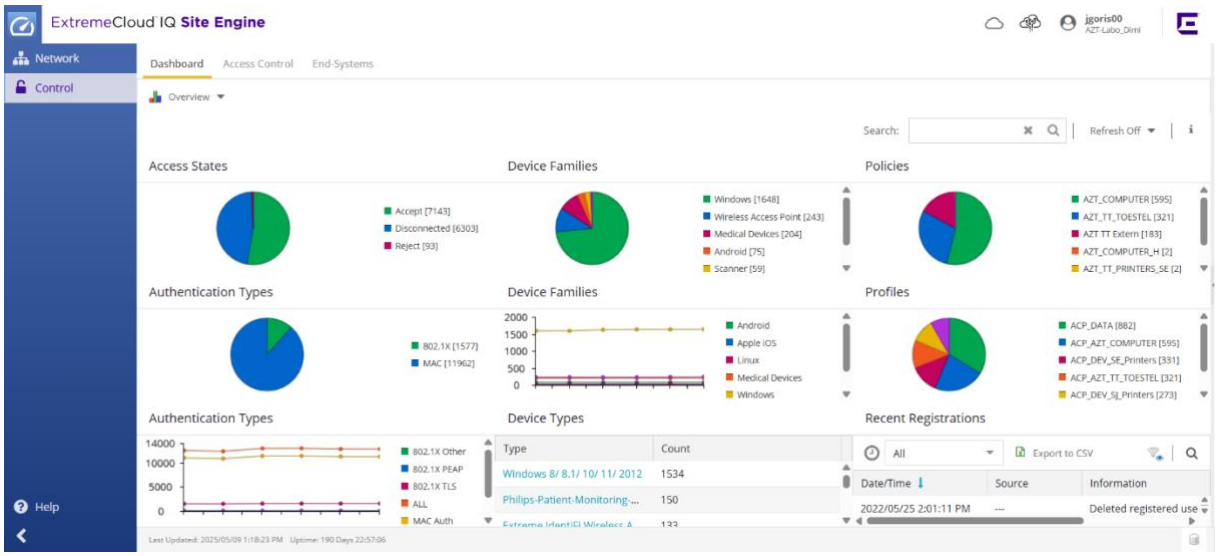
Concreet wil dit zeggen dat we op svkali een subinterface aanmaken, bijvoorbeeld eth0.123 voor VLAN 123, en vervolgens VLAN 123 ‘tagged’ aanbieden op poort 1/48 van de core switch.

Op die manier kunnen we via een statisch IP toch toegang krijgen tot eender welk VLAN in het netwerk, omdat deze allemaal gekend zijn op de core switches, wat bij edge switches niet altijd het geval is.

NAC & Pythonscript

NAC, of Network Access Control, is een technologie die gebruikt wordt om de toegang tot een netwerk te controleren. Dit doet het op basis van het MAC-adres van een toestel, om dit zo af te toetsen op een lijst van policies of rules zoals een firewall dat doet.

Bij het AZ Turnhout maakt men gebruik van Extreme Networks als network vendor. De NAC-oplossing die hiermee gepaard gaat is ExtremeControl. Deze tool is te vinden binnen de ExtremeCloud IQ Site engine.



Figuur 5-10: Dashboard ExtremeCloud IQ Site Engine (Network Access Control)

Deze tool is een groot puzzelstuk in de oplossing. Door in deze tool configuraties te gaan maken bepalen we waar onze svkali in het netwerk terecht komt.

Dat is maar één stuk uit de puzzel. De NAC gaat ons toegang geven tot een bepaald VLAN op basis van ons MAC-adres, maar hoe gaan we dat MAC-adres aanpassen? We kunnen uiteraard niet hetzelfde MAC-adres van onze server meerdere malen in de NAC gaan configureren, dit zou voor problemen zorgen.

We hebben dus voor elk VLAN dat we willen scannen een ander MAC-adres nodig, maar zoveel netwerk interfaces, kabels en switchpoorten hebben we natuurlijk niet.

Om dit te realiseren gaan we het MAC-adres van onze ‘scanninginterface’ spoofen. We kunnen via een bash commando (sudo macchanger) het MAC-adres dat we van onze vendor hebben gekregen op onze netwerk interface zelf gaan aanpassen. Dit manueel doen kost echter veel tijd en heeft een grotere kans op fouten. Daarom heb ik hier een script voor geschreven.

Welke MAC-adressen kiezen we dan? We kunnen in theorie zelf willekeurige MAC-adressen gaan verzinnen, en de kans dat we er eentje kiezen dat al bestaat in ons netwerk is klein, maar met tussen de 3- en 7000 apparaten ook niet heel klein.

Hiervoor kunnen we ‘locally administered MAC-adresses' gaan gebruiken. Dit zijn MAC-adressen die niet worden uitgedeeld door fabrikanten, en gebruikt kunnen worden om het originele MAC-adres te gaan overschrijven. Er zijn vier ‘ranges’ van deze lokale MAC-adressen beschikbaar.

- X2:xx:xx:xx:xx:xx
- X6:xx:xx:xx:xx:xx
- XA:xx:xx:xx:xx:xx
- XE:xx:xx:xx:xx:xx

Uit deze ranges kunnen we vrij een MAC-adres kiezen, om ze lokaal (in ons netwerk) te gaan gebruiken.

Om het spoofen van het MAC-adres van de scanninginterface te vergemakkelijken heb ik een Pythonscript voorzien. Dit script zal een lijst van VLANs voorstellen, waaruit we eentje kunnen kiezen. Achterliggend zal het python script het MAC-adres van onze interface aanpassen naar eentje dat voor dat VLAN in de NAC geconfigureerd staat.

Het script zal eerst de vraag stellen aan de gebruiker om een VLAN te kiezen uit de lijst.

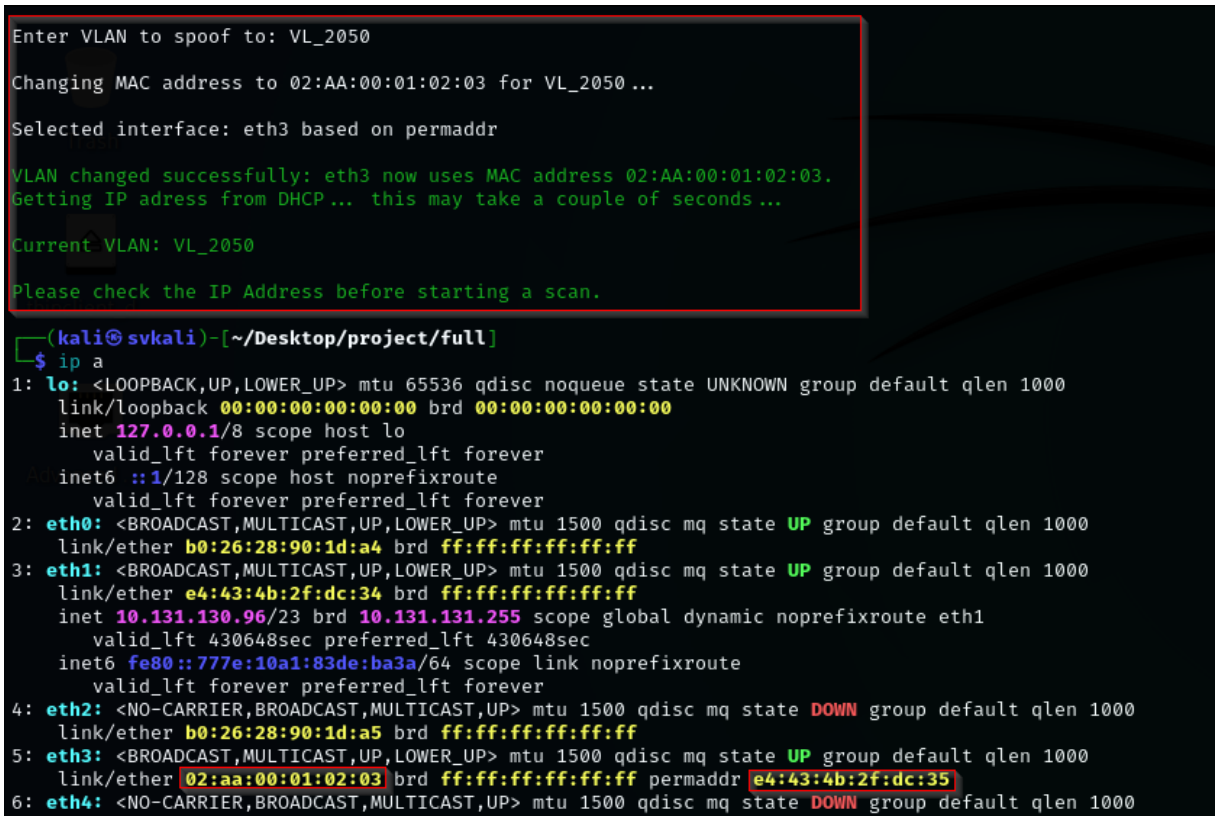
```
    # Display the available options immediately on startup
    print("These are the possible options:")
    for vlan in mac_addresses.keys():
        print(f"  - {vlan}")

    # Prompt user to select the VLAN name or ID
    vlan_choice = input("\nEnter VLAN to spoof to: ")
```

Nadien gaan we nakijken of de keuze van de gebruiker wel degelijk in onze lijst van opties zit.

```
if vlan_choice in mac_addresses:
    new_mac = mac_addresses[vlan_choice]
    print(f"\nChanging MAC address to {new_mac} for {vlan_choice}...")
```

Nu dat we van de gebruiker weten tot welk VLAN hij graag toegang wil, kunnen we ons MAC-adres gaan aanpassen. We kunnen de interface gaan selecteren op naam (bijvoorbeeld eth0), maar dit kan soms wijzigen. Dit is statisch geconfigureerd post-installatie, maar voor de zekerheid selecteren we de juiste interface op basis van zijn ‘permaddr’, oftewel zijn standaard ‘ingebakken’ (origineel) MAC-adres.

Zelfs wanneer we zelf het MAC-adres spoofen, zal het permaddr altijd in de output van het ‘ip a’ commando te vinden zijn:



Figuur 5-11: Resultaat script na selecteren van VLAN, op basis van MAC-adres

We selecteren dus interface eth3 op basis van zijn ‘permaddr’ of permanent adres. Nu weten we zeker dat wanneer we het Pythonscript het MAC-adres laten aanpassen op basis van naam (eth3), dit de juiste interface is die op de NAC-poort is aangesloten.

```
#Use the permaddr to find the interface
  permaddr = "e4:43:4b:2f:dc:35"
   interface = get_interface_by_mac(permaddr)
   if not interface:
      interface = get_interface_by_permaddr(permaddr)
```

Nu we de juiste interface naam hebben, en het juiste mac adres, kunnen we het MAC-adres aanpassen en een DHCP request maken.

```
  if interface:
      change_mac(interface, new_mac)
      time.sleep(1)
      os.system("dhclient >/dev/null 2>&1")
      time.sleep(4)
      new_ip = get_ip(interface)
      print(Fore.GREEN + f"\nCurrent VLAN: {vlan_choice}")
      #print(Fore.GREEN + f"New MAC Address for {interface}: {new_mac}")
      print("\nPlease check the IP Address before starting a scan.")
      #print(Fore.GREEN + f"Current IP Address: {new_ip}")
   else:
      print(Fore.RED + "Could not find an interface to spoof! Configured permaddr does not match the
permaddr of {interface}")
else:
   print(Fore.RED + "Invalid VLAN choice.")
```

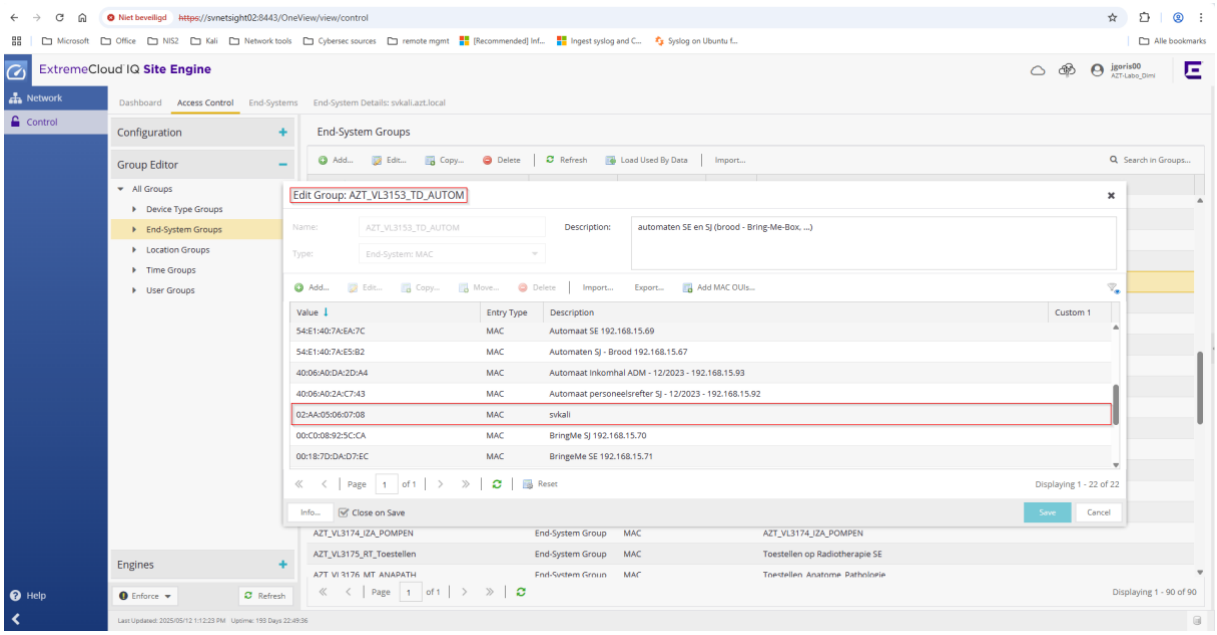
Bovenstaande if statement roept volgende ‘change_mac’ functie om het MAC-adres aan te passen:

```
def change_mac(interface, new_mac):
    if interface:
        print(f"\nSelected interface: {interface} based on permaddr")
        os.system(f"ifconfig {interface} down")
        os.system(f"ifconfig {interface} hw ether {new_mac}")
        os.system(f"ifconfig {interface} up")
        time.sleep(3)
        #os.system("dhclient")
        print(Fore.GREEN + f"\nVLAN changed successfully: {interface} now uses MAC address
            {new_mac}.")
        print(Fore.GREEN + "Getting IP adress from DHCP... this may take a couple of seconds...")
    else:
        print(Fore.RED + "No valid interface found!")
```

Resultaat

Wanneer we het script, NAC en de scanning tools bundelen krijgen we volgende oplossing. Laten we stellen dat we het VLAN ‘VL3153_TD_AUTOM’ willen gaan scannen, een VLAN waarin bijvoorbeeld automaten en bankcontact terminals geplaatst worden.

Allereerst moeten we kijken of we een MAC-adres uit ons script hebben geconfigureerd in de NAC.



Figuur 5-12: Configuratie van svkali met een bepaald MAC-adres in de NAC

Dit komt overeen met een entry in de lijst in ons Pythonscript:

```
    "TEST": "02:AA:01:02:03:04",
    "VL_3222_laadpaal": "02:AA:02:03:04:05",
    "VL_3171_CSA": "02:AA:03:04:05:06",
    "VL_1008_Data": "02:AA:04:05:06:07",
    "VL_3153_td_automaten": "02:AA:05:06:07:08",
    "VL_tbd_6": "02:AA:06:07:08:09",
```

Figuur 5-13: Lijst combinaties VLAN-MAC adres

Wanneer we het Pythonscript uitvoeren en kiezen voor VL_3153_td_automaten zien we dat we van de NAC een IP-adres krijgen voor dit VLAN.

```
┌──(kali㉿svkali)-[~/Desktop/project/full]
└─$ sudo python3 4_mac_spoofer.py
These are the possible options:
  - VL_2050
  - TEST
  - VL_3222_laadpaal
  - VL_3171_CSA
  - VL_1008_Data
  - VL_3153_td_automaten
  - VL_tbd_6
  - VL_tbd_7
  - VL_tbd_8
  - VL_tbd_9
  - VL_tbd_10
  - VL_tbd_11
  - VL_tbd_12
  - VL_tbd_13
  - VL_tbd_14
  - VL_tbd_15
  - VL_tbd_16
  - VL_tbd_17
  - VL_tbd_18
  - VL_tbd_19
  - VL_tbd_20
  - VL_tbd_21
  - VL_tbd_22
  - VL_tbd_23
  - <naam-vlan>
  - VL_tbd_24

Enter VLAN to spoof to: VL_3153_td_automaten

Changing MAC address to 02:AA:05:06:07:08 for VL_3153_td_automaten ...

Selected interface: eth3 based on permaddr

VLAN changed successfully: eth3 now uses MAC address 02:AA:05:06:07:08.
Getting IP adress from DHCP ... this may take a couple of seconds ...

Current VLAN: VL_3153_td_automaten

Please check the IP Address before starting a scan.
```

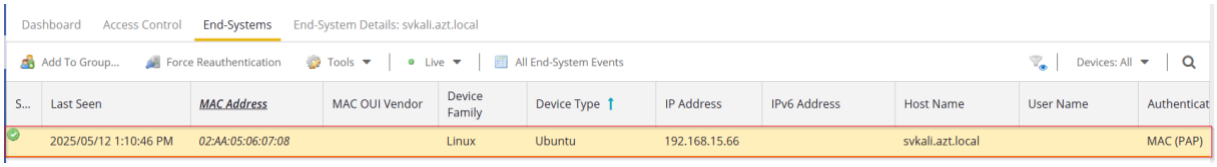
Figuur 5-14: Uitvoeren en gebruiken van Pythonscript, selectie VLAN 3153

Resultaat IP-adres:

```
5: eth3: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc mq state UP group default qlen 1000
    link/ether 02:aa:05:06:07:08 brd ff:ff:ff:ff:ff:ff permaddr e4:43:4b:2f:dc:35
    inet 192.168.15.66/27 brd 192.168.15.95 scope global dynamic eth3
```

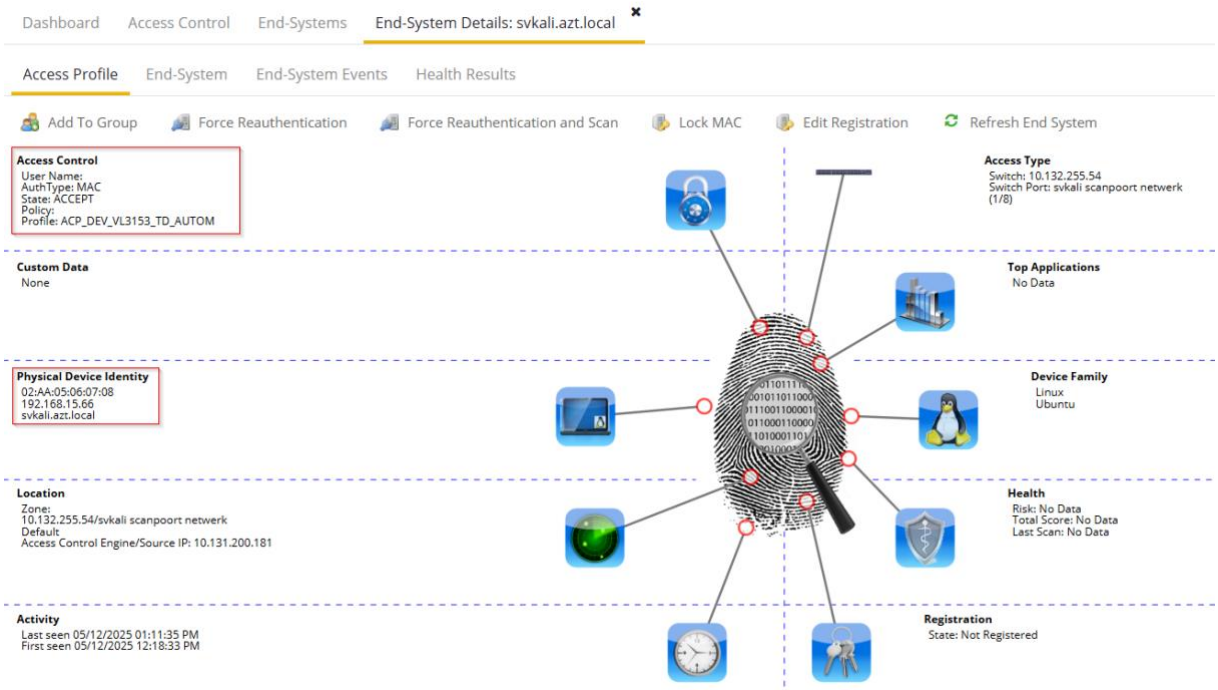
Figuur 5-15: eth3 krijgt IP-adres in correcte VLAN

Resultaat NAC:



Figuur 5-16: NAC geeft end-system weer met nieuwe MAC adres, nieuwe IP-adres en naam svkali

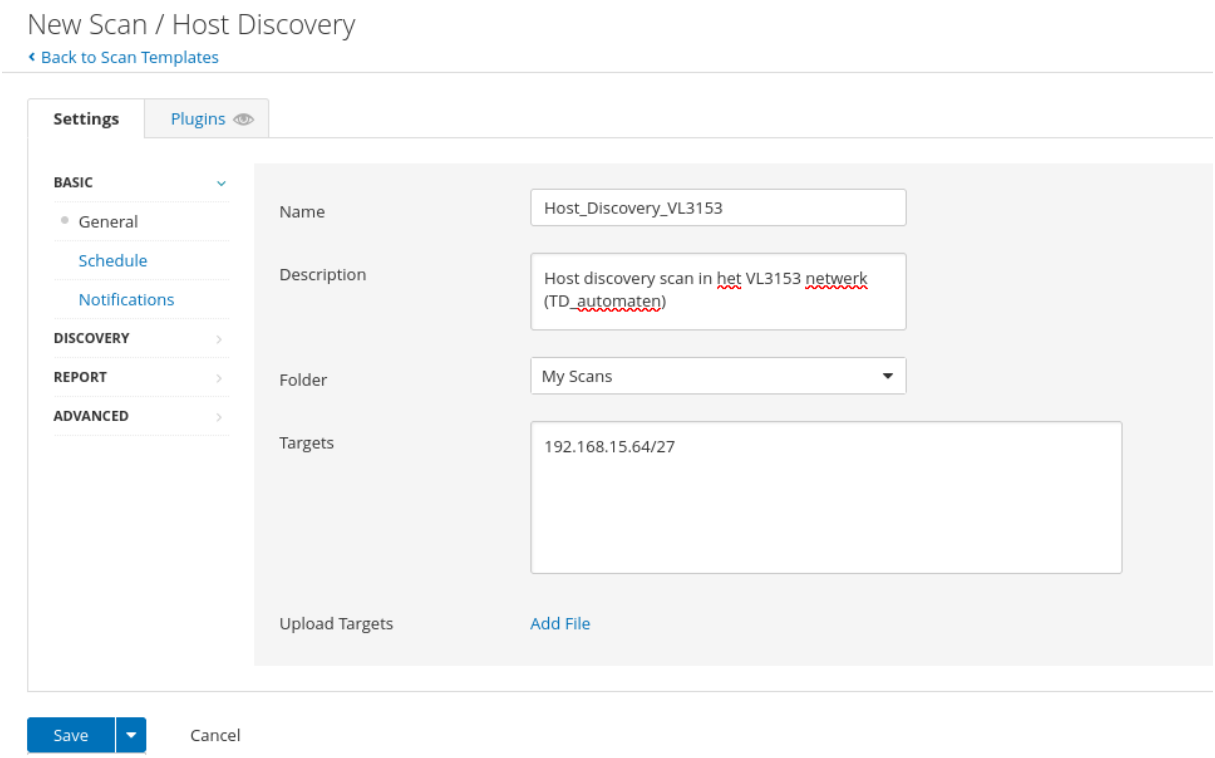
We zijn er op dit punt zeker van dat svkali successvol met het netwerk is verbonden met IP-adres 192.168.15.66. We zien ook het MAC-adres waarop de NAC zijn beslissing heeft gemaakt. Dit is het nieuwe MAC-adres van eth3 dat is toegewezen door ons Pythonscript.



Figuur 5-17: End-system Details van svkali, die verbonden is op het azt.local netwerk

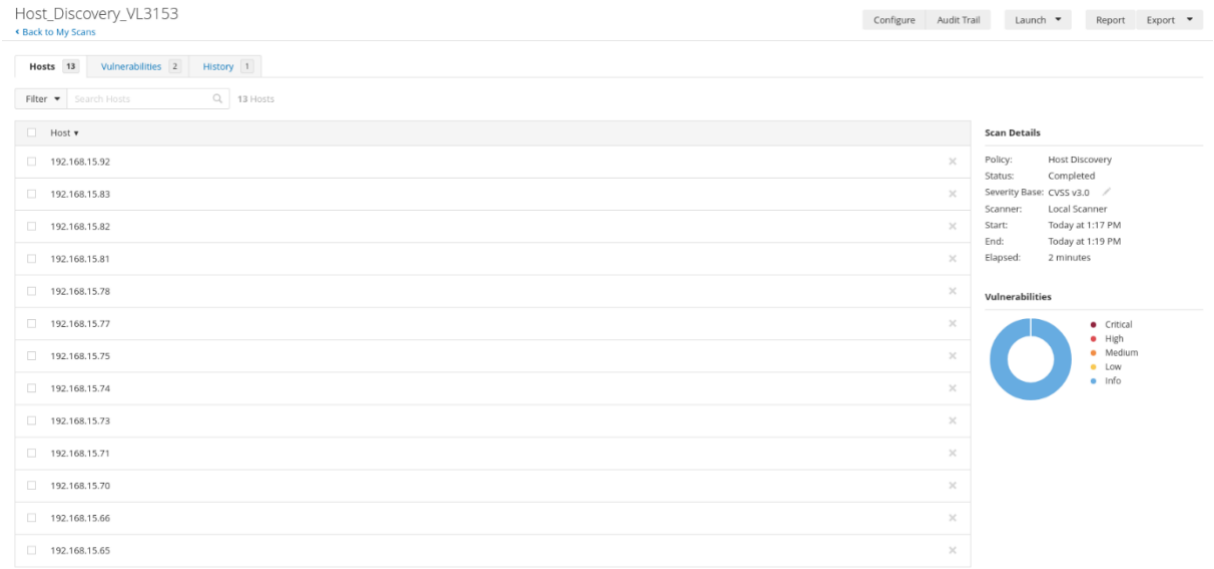
Op bovenstaande figuur zien we enkele details die de NAC ons kan vertellen over het systeem dat verbonden is. We zien op welke switch deze verbonden is, de status van de NAC, het nieuwe MAC-adres en IP-adres, en welk type device dit is. Deze functie geeft ons veel informatie over de apparaten in ons netwerk.

Nu we door het Pythonscript en de NAC in het juiste VLAN zijn geplaatst kunnen we een test uitvoeren door een discovery scan via Nessus te doen:



Figuur 5-18: Configuratiescherm Host Discovery scan in Nessus

Resultaat host discovery scan via Nesus:



Figuur 5-19: Resultaat Host Discovery Scan van Nessus na verplaatsen van VLAN via NAC en Pythonscript

We zien dus dat we via het Pythonscript en de NAC succesvol naar het gewenste VLAN zijn verplaatst en hier een host-discovery scan kunnen uitvoeren. Dit is een teken dat we ook verder kunnen gaan met vulnerability scanning. Deze vulnerability scanning is in detail gedocumenteerd in een interne handleiding.

5.1.3. Documentatie & handover

Om het eerder beschreven systeem en de oplossing voor de opdracht goed te documenteren, heb ik een handleiding opgesteld.

Deze handleiding bevat een gedetailleerde beschrijving van alle stappen die nodig zijn om gebruik te maken van het systeem. Er wordt uitgelegd hoe men verbinding kan maken met de scanning server via RDP of SSH, waar het Python-script te vinden is dat gebruikt wordt om van VLAN te wisselen, hoe dit script gebruikt moet worden, en men het resultaat ervan kan controleren.

Daarnaast worden duidelijke instructies gegeven over hoe men de NAC-configuratie kan bijwerken om de scope van het systeem uit te breiden. Concreet betekent dit dat er stappen zijn toegevoegd die tonen hoe de server aan extra VLANs kan worden gekoppeld in de NAC, wanneer men een nieuw segment wil scannen dat nog niet geconfigureerd is. Ook wordt uitgelegd hoe het script aangepast kan worden om met deze wijziging om te gaan, zodat de verplaatsing naar het nieuwe VLAN ook automatisch verloopt.

Verder bevat de handleiding voorbeeldconfiguraties voor bijvoorbeeld Nessus en OpenVAS, met screenshots van instellingen en resultaten, zodat gebruikers weten wat ze kunnen verwachten.

Op aanvraag is de interne handleiding beschikbaar. Eventuele scanresultaten zijn uit deze versie verwijderd, aangezien ze mogelijks kwetsbaarheden bevatten van specifieke systemen binnen de organisatie.

5.2. Migratie Docker SFTP server

In dit hoofdstuk wordt de migratie van SvDockerHost01 uitgeschreven. Dit is een extra projectje dat er in de loop van de stage is bijgekomen.

5.2.1. Initiële ontdekking

Tijdens een overleg met Hans over mijn hoofdproject, het opzetten van een systeem om vulnerabilties in kaart te brengen, kwam ter sprake dat hij graag zou willen weten hoeveel vulnerabilties er wel niet op de ‘docker host’ zaten. Met de ‘docker host’ bedoelde hij svdocker01, een Ubuntu server die zich in het FTP_DMZ VLAN situeert.

Deze docker host werd vermoedelijk gebruikt voor SFTP. Ik zeg vermoedelijk, omdat deze een zeven-tal jaar geleden is opgezet geweest door een ex-medewerker. De server is nooit gedocumenteerd, en in het huidige infra-team is geen kennis van Docker. Bij wat rondvragen werd deze een beetje gezien als de doos van Pandora. Niemand durfde hier iets mee te doen omdat ze er weinig tot geen kennis van hadden en ook niet goed wisten waar deze SFTP-server juist voor gebruikt werd.

Aangezien ik best wat kennis heb van Docker en hier graag mee werk, leek dit mij een uitgelezen kans om deze kennis hier te gebruiken. Na het uitzoeken van de credentials van de server, ben ik aan de slag gegaan met een authenticated vulnerability scan.

Deze zag er niet al te rooskleurig uit. Er kwamen maar liefst 14 critical, 68 high, 47 medium, 5 low en 113 informational vulnerabilties uit.

Nadien ben ik aan de slag gegaan met het onderzoeken en documenteren van de huidige situatie.

5.2.2. Beoordeling huidige situatie

SvDockerHost01 draait op een Ubuntu 18.04-server. Deze was destijds up-to-date, maar dateert van april 2018 en is inmiddels end-of-life sinds mei 2023. De server is bovendien nooit onderhouden: er zijn geen OS-updates of package-updates uitgevoerd sinds de initiële installatie. Ook de Docker-versie zelf liep al een tiental versies achter.

Een groter probleem zit echter in de containers zelf. Er draaien vijf SFTP-containers en één Watchtower-container. Watchtower is een tool die andere containers monitort en automatisch update wanneer er een nieuwe versie beschikbaar is. Uit mijn onderzoek bleek echter dat de SFTP-containers gebaseerd waren op een lokaal gebouwde Docker image, die uiteraard niet beschikbaar is op Docker Hub.

Bij het controleren van de logs viel op dat Watchtower voortdurend op zoek was naar een container met de naam ‘sftp’, maar er geen vond, omdat die container gebaseerd was op een lokale build. Dit was dus een eerste duidelijke misconfiguratie. Wat wel opviel, was dat de Watchtower-container nog maar anderhalf jaar oud was, wat suggereert dat er relatief recent nog iemand naar dit systeem heeft gekeken.

Ik heb de Watchtower-container gestopt en verwijderd om de constante foutmeldingen in de logs te verminderen.

Bij het analyseren van de gebruikte Dockerfile bleek dat de containers gebruikmaakten van de phusion/baseimage. Dit is een op Ubuntu gebaseerde image, geoptimaliseerd voor gebruik in Docker. De versie die hier gebruikt werd, is gebaseerd op Ubuntu 16.04, dat al sinds 2021 end-of-life is.

Na wat meer onderzoek blijkt dat de ex-collega gebruik heeft gemaakt van volgende GitHub repository, en hier een eigen draai aan heeft gegeven. Dit was een goede vondst, aangezien deze wel wat beter gedocumenteerd is. Wel is deze ook verouderd, en zijn er geen updates aan gebeurd.

https://github.com/MarkusMcNugen/docker-sftp

Samengevat: we hebben een Ubuntu 18.04 host, met daarop vijf Ubuntu 16.04 containers die allemaal publiek toegankelijk zijn via het internet. In de authenticatielogs zijn dan ook talloze brute-force pogingen terug te vinden.

Daarnaast viel op dat er in de bestandsstructuur veel overbodige configuratiebestanden en testconfiguraties aanwezig waren. Deze hoeven dan ook niet mee gemigreerd te worden naar de nieuwe server.

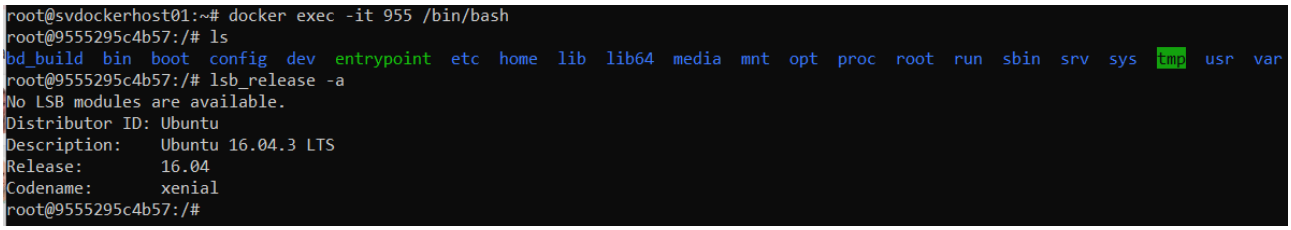
Bewijsstukken:

```
root@svdockerhost01:~# lsb_release -a
No LSB modules are available.
Distributor ID: Ubuntu
Description:    Ubuntu 18.04.6 LTS
Release:        18.04
Codename:       bionic
root@svdockerhost01:~#
```

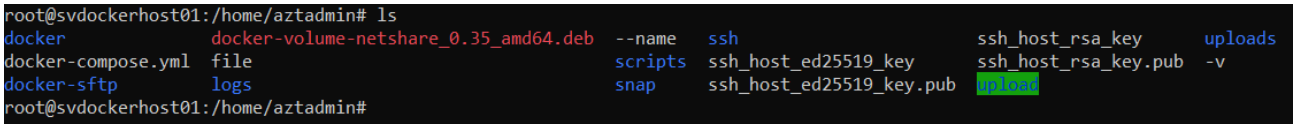
Figuur 5-20: OS versie svdockerhost01

```
root@svdockerhost01:~# docker ps -a
CONTAINER ID   IMAGE     COMMAND         CREATED       STATUS         PORTS                                          NAMES
9555295c4b57   sftp      "/entrypoint"   3 years ago   Up 14 months   0.0.0.0:2234->22/tcp, :::2234->22/tcp          intelligent_saha
6c1aafe44809   sftp      "/entrypoint"   3 years ago   Up 14 months   0.0.0.0:2231->22/tcp, :::2231->22/tcp          blissful_beaver
911a2e59c9b9   sftp      "/entrypoint"   5 years ago   Up 14 months   0.0.0.0:2233->22/tcp, :::2233->22/tcp          wonderful_mahavira
8d982fc3fda2   sftp      "/entrypoint"   5 years ago   Up 14 months   0.0.0.0:2232->22/tcp, :::2232->22/tcp          mystifying_robinson
be310234ea38   sftp      "/entrypoint"   6 years ago   Up 14 months   0.0.0.0:2230->22/tcp, :::2230->22/tcp          elated_goldwasser
root@svdockerhost01:~#
```

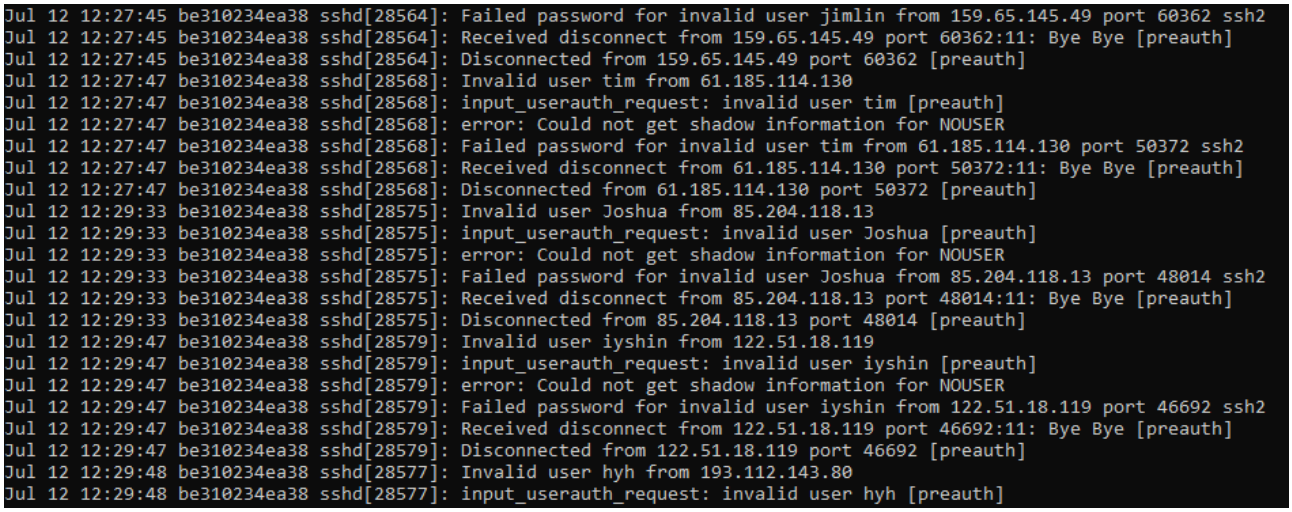
Figuur 5-21: Containers op svdockerhost01



Figuur 5-22: OS versie van image. Screenshot genomen binnen in container (SFTP-server)



Figuur 5-23: Originele bestanden op svdockerhost01



Figuur 5-24: Brute force pogingen op SFTP-server

De conclusie die we kunnen trekken is dat de huidige svdockerhost01 erg verouderd is en aan vervanging toe is. Deze server wordt gebruikt als doorgeefluik, en er zijn dus geen bestanden langdurig aanwezig. Toch kan deze server mogelijks als toegangspunt in het netwerk gebruikt worden, aangezien deze vanop het internet bereikbaar is.

5.2.3. Migratie plan

Na het onderzoeken van de server, wat nodig was omdat er geen documentatie aanwezig is buiten de GitHub repository, ben ik aan de slag gegaan met het uitdenken van een migratieplan. Om de server te kunnen migreren was het nodig om eerst een goed beeld te krijgen van de huidige actieve configuratie, en de onnodige bestanden eruit te filteren.

Zoals in de analyse beschreven is heb ik ervoor gekozen om opnieuw voor een ‘reguliere’ Docker installatie te kiezen, maar deze keer wel met gebruik van docker-compose. Dit maakt het een stuk eenvoudiger om de gehele stack te configureren, zonder dat elke sftp-server afzonderlijk gestart moet worden met de juiste parameters.

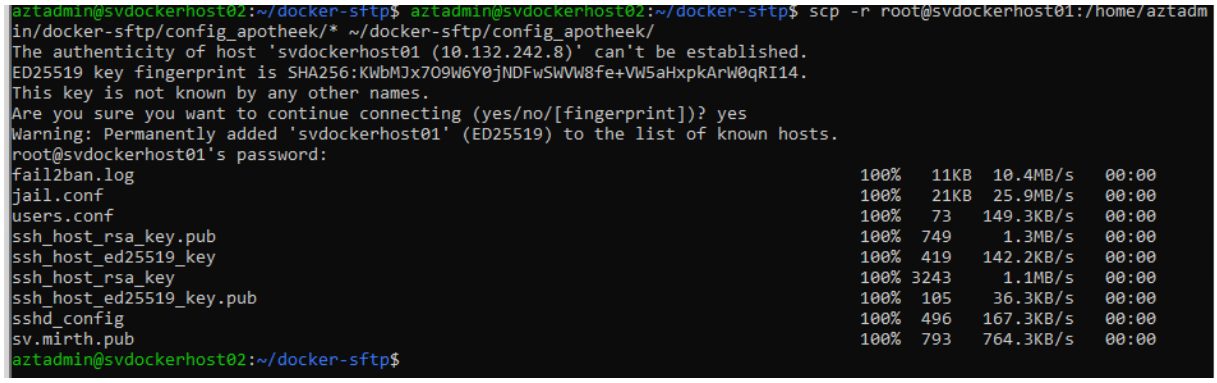
Ook is het zo een stuk eenvoudiger om de servers te beheren en bijvoorbeeld te herstarten. Ook is een docker-compose file makkelijk te documenteren.

De eerste stap die ik heb genomen was het installeren van een nieuwe server. Op de nieuwe server heb ik gekozen voor Ubuntu 24.04, de huidige LTS-versie. Deze heb ik nadien voorzien van Docker, docker-compose, en enige post-installatie configuraties (netwerk, NTP, ...)

Figuur 5-25: SSH banner op svdockerhost02, met warning

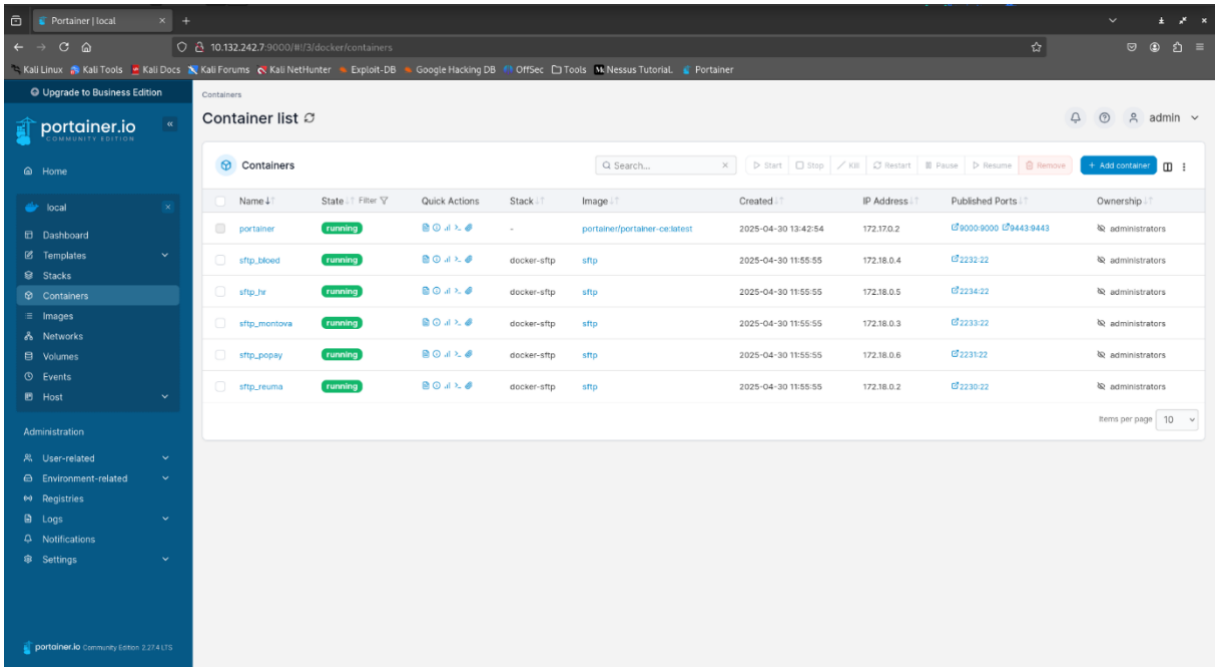
Nadien heb ik enkel de nodige configuratiefiles gekopieerd. Deze configuratiefiles bevatten de users.conf en sshd configuratie die SFTP nodig heeft. Wanneer we de server willen overzetten, is het belangrijk dat de servers die gebruik maken van deze SFTP-server niet door hebben dat er een nieuwe server is. We kunnen hun configuratie namelijk niet aanpassen. Het zou een 1:1 kopie moeten zijn, die een stuk veiliger en up-to-date is. Om dit te bekomen heb ik de /config_XXX mappen gekopieerd naar de nieuwe server via SCP.

Dit om er zeker van te zijn dat ook alle SSH-key configuratie 1:1 is overgenomen.



Figuur 5-26: Kopiëren bestanden van svdockerhost01 naar svdockerhost02

De laatste stap van het migratieplan is om een Portainer instantie te installeren, zodat de huidige stack makkelijk te configureren is. Op deze manier kunnen de collega's die geen of weinig Docker kennis hebben toch makkelijk de servers herstarten, de logs nakijken, de servers 'betreden' alsof het normale Linux servers zijn, enzoverder.



Figuur 5-27: Container list portainer

5.2.4. Verbeteringen

De verbeteringen zijn simpel op te lijsten. We hebben:

- Nieuwere OS
- Nieuwere base image -> containers
- Nieuwere Docker versie
- Docker-compose
- Documentatie
- Portainer voor makkelijke management

5.3. Operationele ondersteuning & incidenten

Dit hoofdstuk beschrijft kort de nevenopdrachten van mijn stage. Dit zijn de taken en opdrachten die ik heb uitgevoerd naast mijn projectwerk. Dit zijn opdrachten die gezien kunnen worden als dagelijks werk, maar waar ik toch erg nuttige zaken uit heb geleerd.

5.3.1. Microsoft Defender

Het grootste stuk van mijn dagelijks werk bestaat uit het opvolgen van het Microsoft Defender-portaal. Bij AZ Turnhout wordt gebruikgemaakt van een volledige Microsoft Defender-suite. Deze bestaat uit verschillende onderdelen, waaronder:

- Microsoft Defender for Endpoint
- Microsoft Defender for Identity
- Microsoft Defender for Cloud Apps
- Microsoft Defender for Office 365

Deze producten werken samen om informatie te verzamelen over wat er gebeurt binnen de omgeving. Op basis van deze informatie genereert Microsoft Defender alerts en incidenten die ons waarschuwen voor mogelijke beveiligingsproblemen of aanvallen.

Defender houdt van alles in de gaten: alles wat gebeurt op endpoints, e-mailbeveiliging via Defender for Office 365, verdachte aanmeldpogingen, verdachte bestanden, enzovoort.

De opvolging van deze incidenten gebeurt samen met onze externe SOC-partner, Secwise. Zij volgen deze meldingen op via ServiceNow (SNOW)-tickets en maken gebruik van playbooks om deze automatisch of handmatig te analyseren en op te lossen. Indien nodig zijn wij de tussenpersoon tussen de eindgebruiker en het incident, om hen zo op de hoogte te stellen.

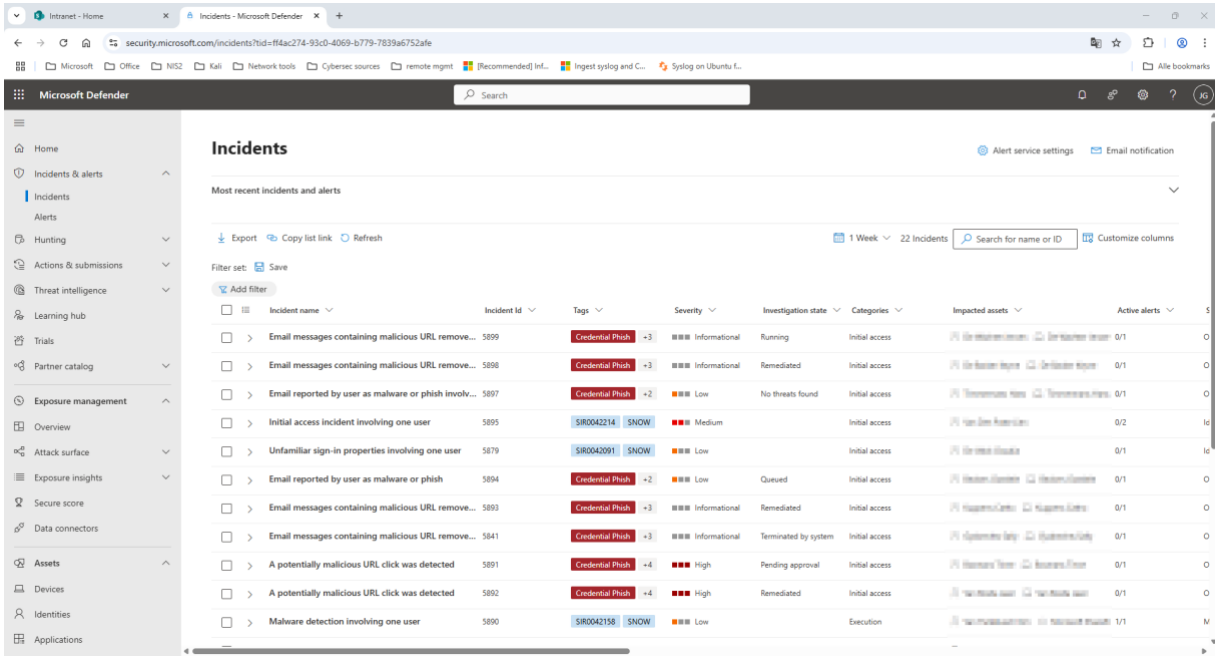
Dagelijks bekijk en beheer ik deze incidenten. Denk aan zaken zoals het blokkeren van afzenders van phishingmails, verdachte domeinen of gevaarlijke URLs, ... Ook gebruik ik Defender om de timeline (processen, logs, ...) van een werkstation (WS) of laptop (LT) na te kijken wanneer er zich problemen met software of connecties voordoen in het ziekenhuis om na te kijken of Defender hier voor iets tussen zit (policies).

Het portaal geeft ook inzicht in de status van onze beveiligingsmaatregelen: op welke toestellen ze actief zijn, en waar nog verbeteringen mogelijk zijn. Deze aanbevelingen gebruiken we dan ook om onze beveiliging verder aan te scherpen.

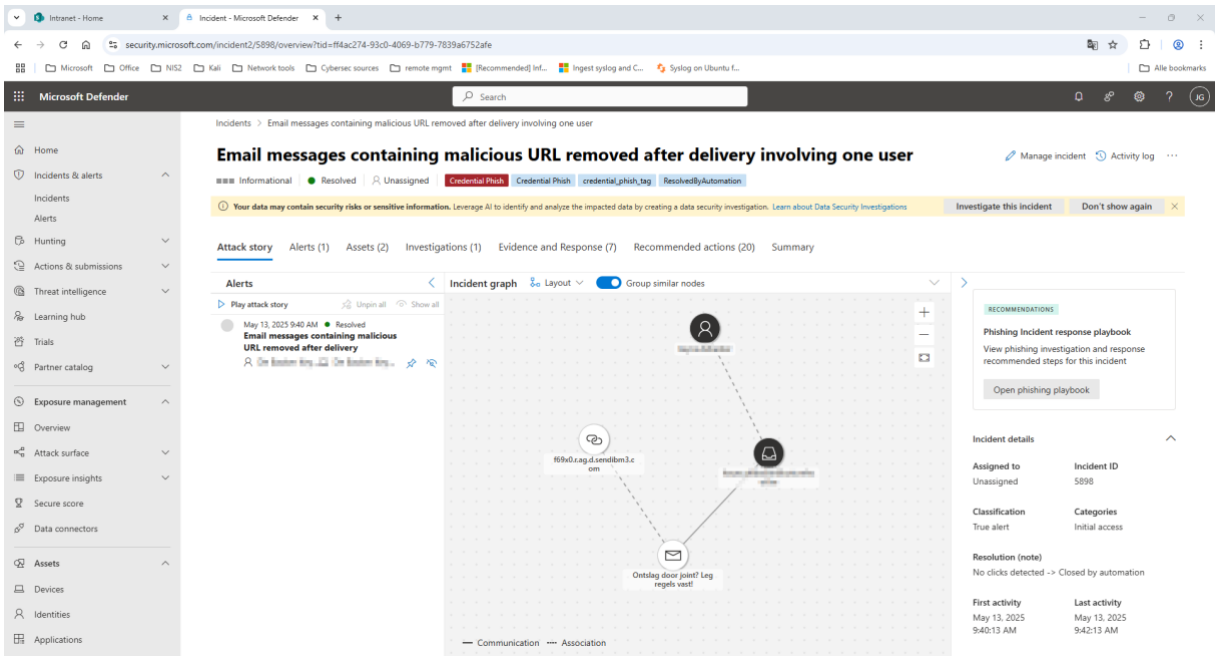
Daarnaast heb ik gebruikgemaakt van de advanced hunting-feature om Attack Surface Reduction-rules (ASR) die momenteel in audit mode staan, te analyseren. Zo kunnen we zien wat de impact zou zijn als we deze regels effectief zouden afdwingen.

Tijdens deze analyses werd ook duidelijk dat er dagelijks meerdere phishingpogingen plaatsvinden. Sommige van deze mails zijn opvallend geloofwaardig en goed opgebouwd.

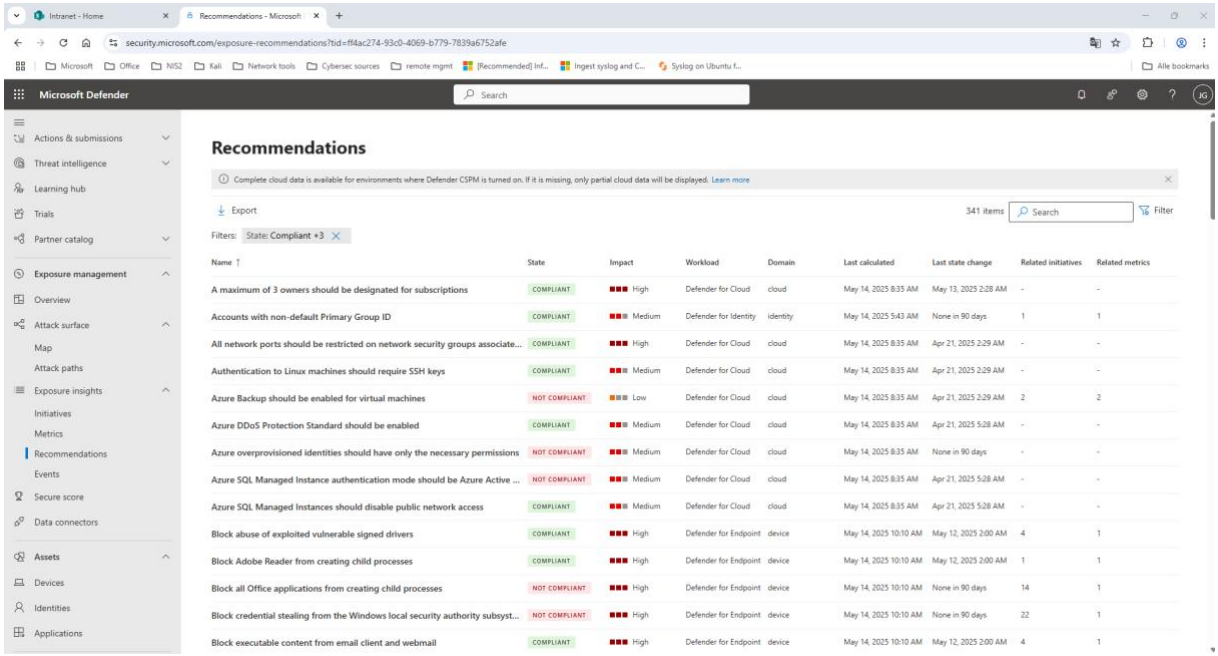
Momenteel worden er via Defender 3485 toestellen, 26 345 identiteiten en 292 cloud-assets actief gemonitord en beschermd.



Figuur 5-28: Incidents tab van Microsoft Defender portaal



Figuur 5-29: Voorbeeld incident in Microsoft Defender portaal



Figuur 5-30: Recommendations onder Exposure management, aanbevelingen van Microsoft

5.3.2. Gebruikersondersteuning

Tijdens mijn stage heb ik vooral in het administratief gebouw gewerkt maar ben ik ook met Lode, ploegleider servicedesk, enkele keren op ‘interventie’ geweest. Deze momenten vond ik erg tof. Zeker in een ziekenhuis kom je zo op ‘gekke’ plekken terecht.

Tijdens week 11 ben ik bijvoorbeeld met samen met Lode een probleem met een label printer gaan oplossen op de dienst sterilisatie, waar we ons eerst helemaal steriel moesten aankleden. Wanneer deze dienst niet kan verder werken kan uiteindelijk OK ook niet meer werken, aangezien zij dan geen steriele sets meer hebben. Zowel sterilisatie als OK krijgen enige prioriteit wanneer zij tickets aanmaken, omdat het ziekenhuis anders stilvalt. Nadien zijn we verdergegaan naar de dienst hartbewaking, om daar een echo-toestel te troubleshooten dat zijn beelden niet meer kon doorsturen naar het medisch beeldbeheersysteem, JiveX. Deze momenten waren niet alleen leerzaam (qua troubleshooting), maar ook erg interessant om zo op de verschillende diensten te komen en met enorm specifieke, medische apparatuur te werken.

Verder hebben we doorheen de stage op andere diensten nog wat support en installaties uitgevoerd.

5.3.3. Samenwerking 3e lijn tickets

Het oplossen van tickets behoort niet tot mijn takenpakket, maar wanneer mijn stagebegeleider Hans bezig is met een interessant ticket probeer ik hier altijd bij te helpen door mee te troubleshooten. Deze tickets zijn vaak security- of email gerelateerd. Het troubleshooten van Active Directory (AD)-kwesties heeft enkele tijd geduurd voor ik in dit verhaal mee was, door de erg ingewikkelde AD structuur en talloze groepsbeleidsregels (GPO’s) die van kracht zijn in onze omgeving.

5.4. Integratie Infoblox BloxOne & Microsoft Sentinel: Research

Binnen AZ Turnhout wordt voor DNS en DHCP gebruikgemaakt van Infoblox. Infoblox is een gespecialiseerde oplossing voor DNS- en DHCP-services binnen grotere IT-omgevingen. In plaats van gebruik te maken van de standaard DNS- of DHCP-roles op een Windows Server, maakt Infoblox gebruik van een dedicated toestel. Het is een volledige en krachtige tool, maar er hangt ook een stevig prijskaartje aan vast. Om die reden kiezen veel (kleinere) organisaties nog steeds voor de Windows Server-oplossing.

Omdat Infoblox een dedicated appliance is, kan het meer requests gelijktijdig verwerken en is het beter geschikt voor grotere omgevingen zoals een ziekenhuis. Een bijkomend voordeel is de integratie met threat intelligence, via Infoblox BloxOne.

Wanneer deze functie geactiveerd is, worden alle DNS-queries eerst via de cloud geleid, waar ze vergeleken worden met threat intel. Zo kunnen bijvoorbeeld malicious domeinen, recent geregistreerde domeinen, of andere verdachte adressen automatisch geblokkeerd worden nog vóór er effectief een verbinding gemaakt wordt. Het blokkeren van malafide DNS-requests is een krachtige manier om het securitypostuur van een organisatie te versterken.

Momenteel is deze functionaliteit enkel actief binnen het ziekenhuis zelf, op het azt.local-netwerk. Infoblox biedt ook een agent aan die op toestellen buiten het interne netwerk (zoals bij thuiswerk of op conferenties) bescherming kan bieden. Hierop zit geen licentie per gebruiker, waardoor de agent makkelijk breed uitgerold kan worden zonder grote bijkomende kost.

Op dit moment draait deze agent enkel op de laptop van mijn stagementor, Hans, als test. In de toekomst is het de bedoeling dat deze verder uitgerold wordt naar meerdere gebruikers.

Een bijkomend voordeel van Infoblox BloxOne is de mogelijkheid om zogenaamde dataconnectors in te stellen. Hiermee kan de data uit BloxOne doorgestuurd worden naar externe systemen zoals een SIEM (Security Information and Event Management), bijvoorbeeld Microsoft Sentinel in ons geval. Hierdoor kunnen de logs van DNS-verkeer extra context geven aan security incidenten. (Schuur, 2021)

Microsoft Sentinel wordt binnen AZ Turnhout beheerd door onze externe SOC-partner, Secwise. Tijdens mijn stage heb ik samen met hen een overleg gehad over hoe deze integratie zou kunnen verlopen. Het opzetten van de ‘dataontvanger’ aan de kant van Sentinel is de verantwoordelijkheid van Secwise. Het instellen van de dataconnector aan de kant van Infoblox was onze verantwoordelijkheid.

Onze lokale Infoblox-toestellen sturen hun data naar de cloudomgeving van BloxOne. In BloxOne kunnen we een connector configureren die deze data doorstuurt naar een lokale syslog-server bij AZT. Deze syslog-server werd eerder al gebruikt voor de integratie van Fortinet-logs en wordt nu hergebruikt.

De syslog-server stuurt op zijn beurt de data door naar Microsoft Sentinel, naar de eerder genoemde dataontvanger.

Tijdens mijn stage heb ik het opzoekwerk en de voorbereiding gedaan voor deze integratie. De eigenlijke implementatie is echter nog niet gebeurd, omwille van een lopend project binnen AZT over het afstappen van het gebruik van wildcardcertificaten (zoals *.azturnhout.be). Voor een veilige overdracht (via TLS) van de logs naar de syslog-server is een specifiek certificaat, bijvoorbeeld voor syslog.azturnhout.be, een betere optie. Deze specifieke certificaten zullen in de toekomst voor meer toepassingen gebruikt worden.

Het gebruik van een wildcardcertificaat is in het algemeen vanwege veiligheidsoverwegingen niet gewenst, en dus is er beslist om de realisatie uit te stellen tot de certificaatstructuur herzien wordt. De enige alternatieve optie zou zijn om de logs via plain TCP (zonder encryptie) te verzenden, maar dat willen we om begrijpelijke redenen vermijden.

De effectieve realisatie van de integratie zal dus naar verwachting gebeuren zodra de huidige certificaten vernieuwd worden.

6. Besluit

Na het afronden van mijn stage bij AZ Turnhout kijk ik met een positief gevoel terug op deze periode. Ik ben erg tevreden met mijn keuze van stageplek, niet alleen vanwege de organisatie, maar ook omwille van de IT-omgeving waarin ik terechtkwam. Een ziekenhuis heeft op IT-vlak een aantal unieke eigenschappen: het aantal endpoints en eindgebruikers, medische toestellen, gevoelige data, combinatie kantoor- en thuiswerk voor administratief personeel, enzoverder. Dit maakt het beheer van informaticasystemen zowel uitdagend als boeiend.

Mijn hoofdproject, het opzetten van een systeem voor interne vulnerabilityscanning in functie van NIS2 heeft me vooral veel inzicht gegeven in hoe je in een productieomgeving met gevoelige toestellen en complexe netwerken toch een betrouwbare en herhaalbare oplossing kan uitrollen. Zeker op vlak van NAC en VLAN-beheer heb ik bijgeleerd. Ook leerde ik dat sommige dingen in de praktijk anders lopen dan op school: niet alles is altijd up-to-date en hyper-modern, en dat is vaak ook niet mogelijk door afhankelijkheden met andere systemen of leveranciers.

Een belangrijke vaststelling is dat je in een grote organisatie vaak afhankelijk bent van anderen. De samenwerking binnen het team verliep goed, en iedereen heeft zijn eigen specialisatie. Maar wanneer je voor je project iets nodig hebt van bijvoorbeeld een collega, kan het soms even duren. Dat is uiteraard logisch aangezien iedereen een eigen takenpakket heeft, maar het heeft er in mijn geval wel voor gezorgd dat bepaalde nevenprojecten, zoals een proof of concept met Allegro, jammer genoeg niet konden doorgaan.

Achteraf bekeken was mijn hoofdproject sneller afgerond dan verwacht. Dat kwam deels doordat de oplossing technisch minder uitgebreid bleek dan voorzien, maar ook doordat er binnen het huidige infrateam weinig ervaring is met Linux. Daardoor was moeilijk in te schatten hoeveel tijd er nodig zou zijn. Persoonlijk had ik graag meer betrokken geweest bij eerstelijnsondersteuning, zoals initieel besproken. Ik vind dat een erg waardevolle leersituatie: zo zie je rechtstreeks welke problemen zich voordoen op de werkvloer, denk aan toepassingen zoals Imprivata voor SSO, profielen op COWs en MUDs, problemen met mailing, KWS of printers. Met je eigen ogen zien hoe een eindgebruiker tegenover een oplossing staat is immers erg leerzaam.

Het is jammer dat ik daar niet nauwer bij betrokken werd. Ik begrijp uiteraard dat in een kritische omgeving niet zomaar elke stagiair verregaande rechten kan krijgen, maar ik geloof dat dit gerust per persoon geëvalueerd kan worden. Zeker omdat ik al een graduaatsdiploma Systeem- en Netwerkbeheer heb, en dus niet volledig nieuw ben in het vakgebied. De eerste weken zouden volgens mij ideaal zijn om mee te lopen met een collega, en nadien stillaan meer verantwoordelijkheden op te nemen. Dit zorgt ook voor een breder takenpakket naast het projectwerk, wat ook de werkdruk voor collega's kan verlichten. De momenten waarop Lode, ploegleider service-desk mij meenam naar interventies vond ik er leerzaam.

Een ander belangrijk leermoment was het verschil in schaal en complexiteit tussen schoolomgevingen en de praktijk. Zaken zoals een AD-omgeving, Office-licenties, Group Policies, en scripting voor zaken zoals HR-AD-synchronisatie zijn in werkelijkheid veel uitgebreider. Ik heb ook geleerd hoe belangrijk automatisering is om een omgeving efficiënt en correct te beheren.

Tot slot vond ik het werken met het Microsoft Defender-portaal bijzonder leerzaam. Ik had hier voordien geen ervaring mee, maar heb gemerkt wat voor krachtige tool het is. Het was indrukwekkend om te zien hoeveel phishingpogingen er effectief dagelijks plaatsvinden, en hoe groot de impact van een geslaagde aanval kan zijn, iets wat ik ook van dichtbij heb meegemaakt.

In het algemeen kijk ik terug op een zeer leerrijke stageperiode binnen een aangename organisatie. Het team was behulpzaam, de sfeer was ontspannen, en er was altijd ruimte voor vragen. Als ik iets anders zou aanpakken, dan zou ik eerder in gesprek zijn gegaan over het takenpakket, zodat dit eventueel nog bijgestuurd kon worden. Ook dat is een leerpunt dat ik meeneem naar de toekomst.

LITERATUURLIJST

ccb. (2023, 03 01). *CyberFundamentals BASIS*. Opgehaald van ccb: https://atwork.safeonweb.be/sites/default/files/2024-05/CyFUN_BASIS_V2023-03-01_N_update%202024.pdf

CCB. (sd). *NIS2*. Opgehaald van Centre For Cybersecurity Belgium.

Hornegold, A. (2024, 5 21). *OpenVAS vs. Nessus - A Comprehensive Analysis*. Opgehaald van intruder.io: https://www.intruder.io/blog/openvas-vs-nessus

nis2.be. (sd). *Cyfun: wat is het en waarom is het belangrijk*. Opgehaald van NIS2: https://nis2.be/waarom-en-wat/cyfun

Schuur, S. (2021, 5 6). *Infoblox Cloud Data Connector Solution for Microsoft Azure Sentinel*. Opgehaald van blogs.infoblox.com: https://blogs.infoblox.com/security/infoblox-cloud-data-connector-solution-for-microsoft-azure-sentinel/

Turnhout, A. (2023, 12 31). *AZ Turnhout in cijfers*. Opgehaald van azturnhout.be: https://www.azturnhout.be/over-azt/jaarverslagen

Turnhout, A. (sd). *Gasthuis in 't groen*. Opgehaald van azturnhout.be: https://www.azturnhout.be/over-azt/gasthuis-t-groen

Turnhout, A. (sd). *kerncijfers*. Opgehaald van azturnhout.be: https://www.azturnhout.be/over-azt/kerncijfers

Turnhout, A. (sd). *Ziekenhuisnetwerk Kempen*. Opgehaald van azturnhout.be: https://www.azturnhout.be/over-azt/ziekenhuisnetwerk-kempen