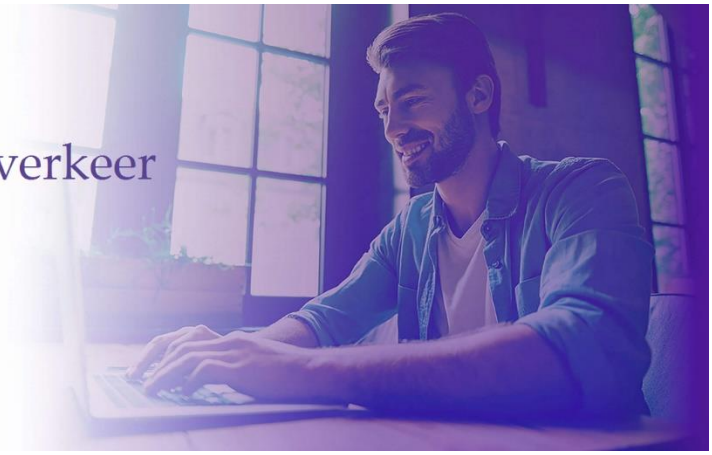
ELEKTRONICA-ICT
Technologiecampus Gent
Gebroeders De Smetstraat 1
9000 Gent

Kwaadaardig internetverkeer onderscheppen

Een firewall ontwikkeld in Python



Kenzo Staelens

Jonathan van Caloen

Jonas Van den Berghe

Michaël Vasseur

Professionele Bachelor Elektronica-ICT

2022-2023

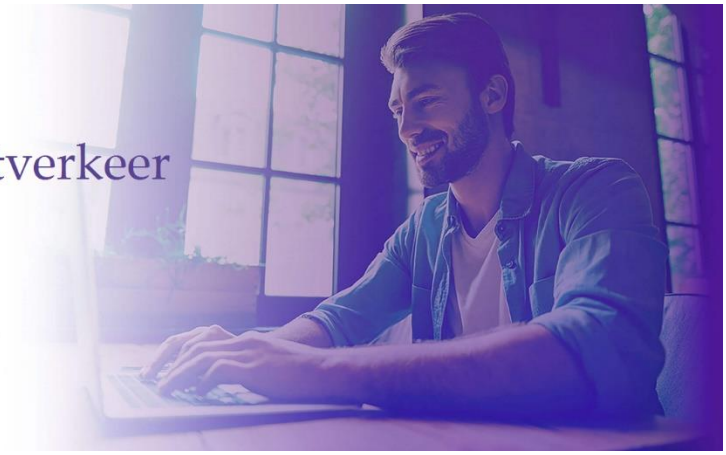
Mentor: Tom Cordemans

Taalmentor: Sabine Martens

ELEKTRONICA-ICT
Technologiecampus Gent
Gebroeders De Smetstraat 1
9000 Gent

Kwaadaardig internetverkeer onderscheppen

Een firewall ontwikkeld in Python



Kenzo Staelens

Jonathan van Caloen

Jonas Van den Berghe

Michaël Vasseur

Professionele Bachelor Elektronica-ICT

2022-2023

Mentor: Tom Cordemans

Taalmentor: Sabine Martens

MALICIOUS INTERNET TRAFFIC INTERCEPTION

K. Staelens, J. van Caloen, J. Van den Berghe, M. Vasseur

This project aims to create a solution for intercepting malicious internet traffic. Where the main focus is ease of use. This is accomplished by dividing the program in two parts, a front-end and a back-end. Which communicates using an API built into the back-end. The front-end is a webpage that retrieves and sends JSON packets using plain Javascript. The back-end consists of the API and the firewall, both written in Python.

The first chapter gives a quick overview of the most common firewall types and how they function. Many firewalls integrate multiple types into one package. This is especially true for modern firewalls, which filter out obvious unwanted traffic before processing it with the more demanding layers. This major performance advantage is why it is also integrated into the project.

The next chapter is meant as an overview of how malware spreads, and why it exists. This information is given for an array of different malware types. Which are chosen based on how prevalent they are in the industry. Gaining a thorough understanding of its functioning is key to preventing infiltration and the further spread of malware. The following chapter dives deeper into efficiently intercepting and blocking malware.

The technical details of the created firewall are in the following chapters. These aim to provide a look at the internal workings of the back-end. The third chapter elaborates on why certain technologies are used. Such as the choice of programming language, UI elements, OS, and more. The specifics on how these technologies are used are in the fourth chapter. Here the back-end coding choices and the module framework are explained. For the front-end, the UI design and UX considerations are shared. And the design of the underlying infrastructure is expanded upon.

Finally, an evaluation of the project and how it evolved is provided. We see this as a successful project with a lot of possibility's for expansion. Some ideas about integrating AI are also shared as a closing thought.

Keywords: Firewall, infrastructure, packet analyser, webserver, malware, artificial intelligence, user friendly interface

Inhoudsopgave

CODEFRAGMENTENLIJST

FIGURENLIJST

TABELLENLIJST

AFKORTINGENLIJST

AI	Artificiële intelligentie
BSOD	Blue screen of death
CSS	cascading style sheets
DDOS	Distributed denial-of-serviceaanval
DoS	Denial-of-service
FTP	File Transfer Protocol
FWaaS	Firewall as a Service
HTML	hypertext markup language
HTTP	Hypertext Transfer Protocol
HTTPS	Hypertext Transfer Protocol Secure
IPS	Intrusion prevention system
JS	Javascript
NGFW	Next-generation firewall
OS	Operating system
OSI-model	Open Systems Interconnection model
SaaS	Software as a Service
URL	Uniform Resource Locator

BEGRIPPENLIJST

Blacklisting	Al het verkeer wordt doorgelaten behalve wat in de regels staat.
Botnet	Een robot netwerk is een verzameling van geïnfecteerde computers waarop malware is geïnstalleerd.
Cross-platform	De mogelijkheid van software, systemen of technologieën om te werken op meerdere besturingssystemen of platforms. Het impliceert dat softwarecode kan worden uitgevoerd op verschillende besturingssystemen zonder dat er wijzigingen of aanpassingen nodig zijn.
Data breach	Verlies van data bij een geslaagde aanval van een dreigingsactor.
Downtime	Diensten of services zijn niet meer bereikbaar na een aanval.
Garbage Collector	Een mechanisme dat gebruikt wordt in programmeertalen met automatisch geheugenbeheer. Geheugen dat niet langer in gebruik is, wordt automatisch gedetecteerd en terug vrij gegeven.
OSI-Model	OSI gestandaardiseerd referentiemodel voor datacommunicatiestandaarden. Hier kunnen de netwerklagen die aan bod komen terug gevonden worden.
Token	Wordt specifiek gegeven aan één gebruiker voor de authenticatie en autorisatie binnen een webapplicatie of API.
Web request	Het proces waarbij een client een verzoek stuurt naar een webserver om informatie op te vragen of een actie uit te voeren. Dit kan bijvoorbeeld via het HTTP-protocol (Hypertext Transfer Protocol).
Whitelisting	Blokkeren van al het netwerkverkeer behalve dat wat in de regels staat.

INLEIDING

Voor dit project wordt er eerst een theoretisch onderzoek gedaan als voorstudie voor het ontwikkelen van een softwarefirewall. De software wordt ontwikkeld in Python. Het project is bedoeld om een beter inzicht te verkrijgen in de werking van malware.

Malware is een samentrekking van de Engelse woorden ‘malicious’ en ‘software’, en wordt vertaald als kwaadaardige software. Het vormt een dagelijkse dreiging voor het internetverkeer en kent vele vormen. Vanuit de vraag naar bescherming wordt er gezocht naar een mogelijke oplossing. Dit kan beantwoord worden vanuit het inzicht dat in de voorstudie wordt verkregen. Dergelijke oplossingen bestaan reeds en worden door verschillende softwarebedrijven op de markt gebracht. Het is dan ook niet de kern van dit project om een betere oplossing te bekomen, maar om een ruimer inzicht in een deelaspect van computerbeveiliging te krijgen.

In het eerste hoofdstuk wordt onderzocht wat malware is. In deze studie wordt besproken welke types er bestaan en wat voor effecten deze hebben bij het geïnfecteerde toestel. Vervolgens worden de bestaande oplossingen besproken. Dit hoofdstuk behandelt firewalls en antivirussoftware.

Uit al deze voorgaande kennis wordt in de volgende hoofdstukken een mogelijke oplossing voorgesteld. Die behandelen de softwarefirewall die ontwikkeld wordt in Python. Het filteren en analyseren van netwerkpakketten moet plaatsvinden op OSI-netwerklaag drie en vier. Ook netwerkpakketten uit bepaalde landen kunnen een verhoogd risico vormen, deze kunnen worden geblokkeerd door het toevoegen van een extra module. Deze module kan de geolocatie bepalen en doorgegeven aan de firewall, die op zijn beurt het pakket al dan niet doorlaat. Een vereiste is dat de software moet draaien op een centrale server die samen fungeert met de firewallsoftware. Hierdoor kan deze later fungeren als een man-in-the-middle. Deze extra module is nodig voor de controle op netwerklaag vier. Dit om HTTPS-pakketten te decrypteren. De firewall moet ook vroegtijdig een DOS-aanval kunnen detecteren en stoppen. Er moet uiteraard interactie zijn tussen de firewall en de eindgebruiker, zodat de gebruiker steeds op de hoogte wordt gesteld van mogelijke aanvallen. De gebruiksvriendelijke interface moet de optie voor een eenvoudig beheer bieden. Ten slotte volgt een voorstel tot een mogelijk vervolg, waar artificiële intelligentie een rol zou kunnen spelen bij de behandeling van netwerkpakketten.

De belangrijkste onderzoeksmethode is literatuurstudie van online bronnen. Het geeft inzicht in de werking van malware en hoe er zich tegen te beschermen. Ook bieden de testen die worden uitgevoerd op de firewall bij de ontwikkeling veel inzicht.

1 FIREWALLS

1.1 DOEL

Apparaten verbonden met het internet zijn steeds kwetsbaar voor dreigingen van buitenaf. Afhankelijk van het doel van de aanval, kan er bijvoorbeeld verlies van data optreden. In vakterminologie spreekt men dan van een data breach. Dit kan heel wat gevolgen met zich meebrengen. Persoonlijke gegevens van klanten kunnen gepubliceerd worden wanneer men niet voldoet aan de eis van de aanvaller. Een onderneming wordt aansprakelijk gesteld voor het verlies van persoonsgegevens, wat ervoor zorgt dat de financiële schade zeer snel kan oplopen. Een ander voorbeeld van een aanval kan zich uiten in het volledig onbereikbaar maken van een netwerk of service. In vakterminologie spreekt men dan van downtime. Hierdoor kunnen klanten geen bestellingen meer plaatsen, of kunnen werknemers hun werk niet meer uitoefenen. Dit kan de onderneming heel wat financiële schade opleveren met een gemiddelde kost van 5600$ per minuut.[1] In het hoofdstuk malware worden de doelen van een aanval verder besproken. Om apparaten die met het internet verbonden zijn te beschermen tegen aanvallen, hebben we firewalls nodig. Het is een eerstelijnsbescherming voor het blokkeren van gevaarlijk of schadelijk netwerkverkeer. Op deze manier worden de dieperliggende netwerkapparaten ook vrijgehouden voor andere taken, waaronder complexere bedreigingen ontdekken. [2] Niet alle malware kan namelijk door de firewall gedetecteerd worden.

1.2 WERKING

Een firewall kijkt naar de eigenschappen van de netwerkpakketten. Eenvoudige firewalls kijken enkel naar de bron en bestemming van zowel IP-adressen als poorten. Waar complexere firewalls ook kijken naar de patronen van ontvangen en verstuurde pakketten en naar de data die in die pakketten zit. Enkel het netwerkverkeer wordt hier gecontroleerd. Wanneer malware via een ander medium het toestel bereikt en geen verdachte pakketten stuurt naar de buitenwereld, worden deze niet gedetecteerd. De manier waarop de detectie gebeurt hangt af van het type firewall.

1.3 TYPES FIREWALL

1.3.1 STATELESS EN STATEFUL

De klassieke firewalls zijn stateless, dit betekent dat ze enkel kijken naar individuele pakketten. Binnen een pakket kijken ze enkel naar de informatie die in de headers van het pakket te vinden is. De bron- en bestemmingsadressen en poorten zijn belangrijke elementen om te beslissen of pakketten al dan niet veilig zijn. De andere parameters moeten manueel ingesteld worden door gebruik te maken van regels. Als een pakket niet aan de regels voldoet wordt het geblokkeerd.

De firewalls die momenteel in productieomgevingen worden gebruikt zijn daarentegen stateful firewalls. Dit houdt in dat het pakket bijgehouden wordt om later te kijken naar patronen in het netwerkverkeer. De mogelijkheid bestaat dus om naar een volledige sessie te kijken. Zo kan een firewall beslissingen nemen op eerder ontvangen verdacht verkeer en die pakketten vroegtijdig tegenhouden zonder nieuwe regels nodig te hebben. [3] [4] Deze functionaliteit loopt parallel met de basisfuncties van een stateless firewall.

1.3.2 STATELESS FIREWALLS

PACKET FILTERING FIREWALL

Dit is een minimale vorm van firewall aangezien de firewall enkel een lijst van regels bevat. Als een pakket binnenkomt bij een packet filtering firewall kijkt die in de geconfigureerde regels en de regels die matchen met de beschrijving van het pakket worden dan gebruikt om een beslissing te nemen. Vaak heeft de beslissing van blokkeren voorrang op het toelaten van een pakket. Als een pakket toch moet worden doorgelaten, dan krijgt de desbetreffende regel een lager ingestelde prioriteit. Een lagere waarde voor prioriteit wil zeggen dat een regel meer doorslag heeft op de finale beslissing of een pakket wordt geaccepteerd of niet. [5] Het filteren van pakketten kan op twee manieren. De eerste is whitelisting, deze blokkeert al het netwerkverkeer behalve dat wat in de regels staat. De tweede is blacklisting, hier gebeurt net het omgekeerde. Al het verkeer wordt doorgelaten behalve wat in de regels staat. Het is altijd veiliger om whitelisting toe te passen.

PROXY FIREWALL

Een proxy firewall werkt als een standaard proxy met geïntegreerde firewall. Zo'n firewall kan dan ook naar data in de pakketten kijken op lagen vijf tot zeven van het OSI-model. Dit kan omdat de proxy zelf ook een certificaat bevat en daarmee een koppeling maakt met de client die een request stuurt en de server dat request ontvangt. Zo kan de proxy de pakketten decrypteren en naar de data in lagen vijf tot zeven kijken. Het grote voordeel van deze firewall is dat een externe netwerkconnectie geen direct contact kan hebben met het netwerk.

1.3.3 STATEFUL FIREWALLS

NEXT-GENERATION FIREWALL

Een next-generation firewall (NGFW) gaat veel verder dan enkel pakketten filteren. Om als next-generation beschouwd te worden, moet de firewall aan een aantal voorwaarden voldoen:

- een slim-aanpassende toegangscontrole gebaseerd op stateful inspectie;
- een geïntegreerd intrusion prevention systeem (IPS);
- mogelijkheid tot upgraden naar nieuwere databronnen;
- URL-filtering gebaseerd op reputatie en geolocatie;
- het controleren en blokkeren van gevaarlijke applicaties;
- detectie over het hele netwerk.

Deze functies zorgen ervoor dat de firewall veel sneller en preciezer kan reageren op mogelijke bedreigingen. Ook zorgt dit ervoor dat een veel kleinere kans bestaat dat de bedreigingen in het lokale netwerk hun functie kunnen uitvoeren.

De implementatie en verdere functies hangen sterk af van product tot product. Verder worden nog twee types next-generation firewalls aangehaald. [5]

THREAT-FOCUSED NEXT-GENERATION FIREWALL

Dit soort firewall houdt zich vooral bezig met bedreigingen detecteren en blokkeren. Dit doet het nog voordat de bedreiging het interne netwerk binnen kan komen. Voordat deze firewall een beslissing neemt worden nog veel meer uitgebreidere testen uitgevoerd dan bij andere firewalls. Om dit te bereiken maakt die onder andere gebruik van een sandbox-omgeving. Dit komt neer op een gemonitorde container die geïsoleerd is van de rest van het systeem. Als die geïnfecteerd wordt door een virus kunnen gepaste maatregelen genomen worden. Hierna wordt de container automatisch verwijderd. Na de procedure in de sandbox wordt een beslissing genomen of het bestand veilig is of niet.

UNIFIED THREAT MANAGEMENT (UTM) FIREWALL

Een UTM doet nog meer dan een next-generation firewall en voert ook de taken uit van een traditioneel antivirusprogramma, waaronder bescherming van individuele apparaten, zoals bescherming tegen dataverlies. De functies kunnen zeer sterk afhangen van de omgeving waarin de UTM-firewall wordt gebruikt. Toch zijn ze makkelijker in gebruik te nemen dan een NGFW. Dit maakt ze interessanter voor kleine tot middelgrote bedrijven.

1.3.4 HARDWARE FIREWALLS

Hardware firewalls zijn firewalls die op basis van vooraf ingebouwde hardwareregels veel sneller kunnen verwerken dan wanneer het op basis van softwareregels moet verwerken. Dit geeft vele voordelen. De eigen hardware en software maakt ze veel veiliger tegen malware die gebruik maakt van fouten in het onderliggende besturingssysteem. Aangezien ze niet door een besturingssysteem worden verwerkt kunnen andere processen moeilijk inpikken op de beslissingen. Vaak loopt dit zelfs over een apart ingebouwde chip en zal de snelheid van de routing niet gehinderd worden door de rekentijd van het checken van pakketten.

Doordat de firewall gecentraliseerd is, is het aanpassen van de configuratie voor de netwerkbeheerder veel gemakkelijker. Dit garandeert dat elk apparaat achter de firewall dezelfde beveiliging krijgt. Ook zorgt dit ervoor dat eindgebruikers en malafide software deze configuraties veel moeilijker kunnen bereiken.

Aangezien de firewall ingebouwd is in de hardware is het wel moeilijker om de fysieke mogelijkheden te upgraden. Daardoor is vaak een volledig nieuwe firewall nodig. Ze hebben hierdoor ook een veel hogere aankoopprijs. De unieke hardware maakt het ook moeilijker om te onderhouden, vaak is dus een specifieke expertise nodig. Als toch updates moeten uitgevoerd worden zal dit gebeuren door het gebruikte besturingssysteem te updaten.

1.3.5 SOFTWARE FIREWALLS

Software firewalls kunnen gedraaid worden op niet gespecialiseerde hardware. Dit kan als één of meerdere processen op het besturingssysteem. Vaak worden deze gebruikt als client based firewalls. Hierbij is te vermelden dat zo'n firewall met root privileges draait. Wanneer malware dus toegang krijgt tot de firewall dan krijgt het ook volledige controle over het netwerkverkeer. Vaak heeft dit type firewall veel standaard ingestelde regels.

Nog een mogelijkheid voor het gebruik van een software based firewall is als service op het toegangspunt tot een netwerk. Voordat dit toegangspunt pakketten kan toelaten moeten deze eerst langs de software passeren. Zo kan dit type toch aan de rand van een netwerk staan.

Dit soort firewall is vaak veel kosteneffectiever dan een hardware firewall omdat de middelen veel makkelijker geschaald kunnen worden. Als de firewall niet genoeg geheugen heeft, kan er simpelweg meer geheugen aan het proces toegewezen worden. Ook is de aankoopprijs veel lager aangezien dit vaak enkel over een softwarelicentie gaat. Het installeren kan volledig van op afstand gebeuren en updates zijn dan makkelijker te installeren.

1.3.6 CLOUD FIREWALLS

Meer en meer delen van organisaties worden naar de cloud verplaatst. Ook de firewall heeft een variant die in de cloud werkt. Wanneer een organisatie services in de cloud heeft kan dit dus een goede optie zijn. Dit kan als off-site virtuele machines of als een SaaS-model. Een firewall-as-a-Service (FWaaS) aanschaffen heeft een aantal unieke voordelen. Zo heeft deze een veel grotere, praktisch onbeperkte, bandbreedtelimiet. Verder kan een FWaaS snel schalen om meer of minder verkeer te kunnen verwerken. Hierdoor kan de startkost zeer klein zijn.

1.3.7 HYBRIDE FIREWALLS

Hybride firewalls zijn een combinatie van meerdere firewalls. Zo kan voor verschillende scenario's op het netwerk een ander type verwerking gebeuren afhankelijk van hoe veilig de verbinding moet zijn. Zo kan in een minder gevoelige regio van het netwerk een minder intensieve firewall gebruikt worden. Hybride firewalls kunnen zo het netwerk veiliger en sneller maken en toch goedkoper zijn.

2 MALWARE

2.1 TYPES

2.1.1 VIRUS

Een computervirus is malware die zich nestelt in andere bestanden zoals bestanden van het operating system (OS). Het is ontworpen om schade toe te brengen aan het systeem en om zichzelf te dupliceren. De verspreiding van virussen gebeurt door overdracht van de besmette bestanden. Belangrijk is dat een virus steeds door iemand geactiveerd moet worden. Het meest voorkomende medium is een e-mail, USB-stick of via bestandoverdracht met het File Transfer Protocol (ftp).

2.1.2 WORMS

In tegenstelling tot virussen, hebben wormen geen menselijke interactie nodig om zich te verspreiden. Ook zijn ze niet afhankelijk van software zoals een virus. Het is malware die in staat is om kopieën van zichzelf te maken en zich meestal via een zwak punt in het systeem te verspreiden. Een zwak punt kan een e-mail, geïnfecteerde websites of andere medium zijn.

2.1.3 TROJANS

Een Trojaans paard of kortweg Trojan slaat op de oorlogsvoering tijdens de Trojaanse oorlog waar een houten paard werd gebruikt om manschappen in te verbergen. Door deze misleiding konden ze de stad binnendringen en veroveren. Een Trojan in de computerwereld is hierop gebaseerd. Door de malware te vermommen als legitieme software kan een kwaadaardige actor een apparaat overnemen. In tegenstelling tot wormen kan een Trojan net als virussen zichzelf niet zelfstandig verspreiden en moet hij door de gebruiker worden geïnstalleerd.

2.1.4 SPYWARE

Spyware is malware die zich op het besmette apparaat verbergt. De spionagesoftware monitort de acties van de gebruiker en verzendt de vergaarde gegevens door naar de kwaadaardige actor. Gestolen gegevens kunnen persoonsgegevens, bankgegevens, e-mailadressen of wachtwoorden zijn. Het stelen van persoonlijke gegevens kan voor verschillende doeleinden worden gebruikt. De meest voorkomende is om geld te stelen.

2.1.5 RANSOMWARE

Ransomware wordt ook wel gijzelsoftware genoemd en is malware die gebruikt wordt als chantagemiddel. Het is malware die het systeem kan encrypteren of waar de toegang kan van worden geblokkeerd. Wanneer een besmet systeem verbonden is met een netwerk, kan het zich heel snel en makkelijk verspreiden. Zo worden meer systemen gegijzeld. Het doel ervan is om geld te vergaren. De kwaadaardige actor kan ervoor kiezen om de bestanden te downloaden alvorens ze te gijzelen. Er wordt gedreigd om de gestolen data publiekelijk te maken. Dit chantagemiddel wordt steeds vaker gebruikt. Wanneer persoonsgegevens publiekelijk gemaakt worden, kunnen ondernemingen aansprakelijk gesteld worden waarbij ze grote financiële schade kunnen oplopen.

2.1.6 ADWARE

Adware of “advertising malware” is malware waarbij een programma ongewenste aanpassingen aan het systeem maakt. Het uit zich voornamelijk in het ongewenst tonen van advertenties. Wanneer men gratis software downloadt, downloadt men daarnaast ook vaak een adware. Dit is volkomen legaal aangezien er een akkoord wordt gegeven vooraleer men de gratis software kan downloaden. Het wordt vaak besproken als een onschuldige soort malware, toch kunnen zij een nog grotere dreiging vormen voor de gegevensprivacy. Het kan doelgerichte advertenties sturen omdat het de onlineactiviteiten kan traceren. Ontwikkelaars van adware verkopen de onlineactiviteiten door aan externe partners. Het is moeilijk om adware te verwijderen omdat het stukje software zich diep in het systeem nestelt.

2.1.7 SCAREWARE

Scareware is malware waarbij een programma de gebruiker probeert aan te zetten om software aan te kopen. Dit door angst of twijfel te zaaien na het tonen van een boodschap. Dit is meestal een waarschuwing van een mogelijke besmetting door een virus. Het programma toont welke software nodig is om het probleem op te lossen. Vaak bieden deze geen oplossing en koopt men opnieuw scareware.

2.1.8 ROOTKITS

Een rootkit is een verzameling van software die wordt gebruikt door een kwaadaardige actor die een systeem is binnengedrongen. Het onderschept en wijzigt processen van het OS. Zo wordt het onder andere gebruikt om andere malware verborgen te houden. Het is zeer moeilijk deze te verwijderen omdat ze diep in het OS worden geïnstalleerd. Eens een systeem is geïnfecteerd kan geen informatie meer vertrouwd worden. Het kan daarom ook jarenlang aanwezig blijven als het niet opgemerkt wordt. Er zijn tools beschikbaar om een rootkit te verwijderen, maar een nieuwe OS-installatie wordt aangeraden als men zeker wil zijn.

2.1.9 BROWSER HIJACKERS

Browser hijackers of “browser omleiding virus” is malware die een webbrowser aantast door instellingen aan te passen zonder de toestemming van de gebruiker. Het verkeer wordt omgeleid naar kwaadaardige websites.

2.1.10 CRYPTOMINERS

Cryptominers ook wel “cryptojacking” genoemd is malware die ontworpen is om geïnfecteerde systemen te gebruiken voor het ontginnen van cryptomunten. Het gebruikt de CPU en het RAM-geheugen van deze systemen, hierdoor vertragen de geïnfecteerde systemen of werken die niet optimaal.

2.1.11 LOGIC BOMBS

Een logic bom is malware die enkel geactiveerd wordt als er aan een bepaalde voorwaarde voldaan wordt. Het bevat een reeks van instructies die worden uitgevoerd na bijvoorbeeld een bepaald tijdstip of wanneer een bedrijf een bepaalde release van software uitbrengt. Het bevat vaak een virus of een worm.

2.2 TEKENEN VAN INFECTIE

Wanneer een systeem geïnfecteerd is met malware, uit zich dit in:
- een traag tot onstabiel systeem;
- ongevraagde vensters, pop-ups;
- Bluescreen Of Death(BSOD);
- verlies van persoonsgegevens of data;
- verlies van valuta.

2.3 GEBRUIK

Malware wordt bijna altijd gebruikt voor criminele doeleinden. Met uitzondering van Adware. Toch ligt ook daar het accent op inkomsten vergaren. In de voorbeelden wordt vaak geld als doel besproken, maar dit hoeft niet altijd het geval te zijn. Een dreigingsactor kan ook als doel hebben om een onderneming te saboteren. Wanneer een dreigingsactor een bepaald doel wil bekomen, zal er gekozen worden voor een daarbij horende vorm van malware. Een aanval wordt zeer goed bedacht alvorens hij wordt uitgevoerd, dit om de slaagkans te vergroten. Ondernemingen worden vaker het slachtoffer van ransomware, waarbij als chantagemiddel het publiek maken van de gestolen data wordt gebruikt. Scareware daarentegen richt zich meer op het individu door angst of twijfel te zaaien, waardoor er software wordt aangekocht.

Een dreigingsactor kan malware gebruiken om een veel groter doel te bereiken en kan systemen infecteren om te gebruiken als een botnet. Dit zijn systemen die als groter geheel worden gebruikt tijdens bijvoorbeeld een distributed denial-of-serviceaanval. Het doel hiervan is om een netwerk of online service ontoegankelijk te maken.

2.4 MEDIUM

Een systeem kan op verschillende manieren geïnfecteerd worden met malware. Vaak gebeurt dit door de gebruiker zelf, maar dit hoeft niet altijd zo te zijn. Een virus wordt overgedragen door een menselijke tussenkomst, waar een worm dit zonder kan. Afhankelijk van wat de dreigingsactor wenst te bereiken, zal een bepaalde strategie worden toegepast. Er zijn verschillende mediums die kunnen leiden tot besmetting:

- een link in een e-mail of chatbericht;
- een link op een website;
- een advertentie;
- een USB-stick;
- een worm kan zich verspreiden doorheen het netwerk.

3 MOGELIJKE OPLOSSINGEN

3.1 CRITERIA

Criteria zijn zeer belangrijk om te kunnen bepalen of een applicatie zijn uiteindelijke doel kan bereiken. Hier worden enkele kenmerken besproken waar het project aan moet voldoen. Een eerste kenmerk is dat de firewall het netwerkverkeer moet classificeren. Dit wordt behaald doordat de gebruiker bepaalde regels kan toevoegen en verwijderen. Omdat er een eenvoudige userinterface wordt voorzien, is het beheer van de firewall dan ook voor iedereen toegankelijk. Met perspectief om de software verder te ontwikkelen, wordt er gekozen om met modules te werken. Een voorbeeld hiervan is het blokkeren van pakketten afkomstig uit bepaalde geografische zones.

3.2 MOGELIJKE OPLOSSINGEN

3.2.1 PROGRAMMEERTAAL

Er zijn verschillende opties betreffende programmeertaal. Belangrijk bij de keuze is dat diegene dat de software ontwikkelen voldoende vaardig zijn in de taal. Een eerste optie is Python. Python is een geïnterpreteerde taal, wat betekent dat het langzamer kan zijn dan gecompileerde talen zoals C of Java. [6] Python is ook zeer goed geïntegreerd in Linux omgevingen. De vele modules die voor deze taal bestaan kunnen de ontwikkeling sterk vooruithelpen.

C# een krachtige cross-platform programmeertaal die wordt gebruikt binnen het .NET-framework. Het is gericht op objectgeoriënteerd programmeren. C# biedt ook zeer goede prestaties door zijn ingbouwde garbage collector. Verder is het ook een zeer gestructureerde taal die het lezen en schrijven makkelijker maakt. [7]

3.2.2 USERINTERFACE

De userinterface kan voorkomen als een applicatie. Firewall software die aangeboden wordt als een applicatie, voorziet zowel de firewall software en de interface als één pakket. De applicatie vorm heeft als nadeel dat het bestuurssysteem afhankelijk wordt. Een andere vorm die gebruikt wordt is een webapplicatie. Hierbij wordt de userinterface ter beschikking gesteld aan de hand van een webpagina. Die wordt gehost op een server en is bereikbaar via het internet. Er zijn veel verschillende manieren om een webpagina te creëren, maar de traditionele manier is het schrijven van hypertext markup language (HTML), cascading style sheets (CSS) en javascript (JS). HTML-bestanden zijn de pagina's waarin alle info staat die nodig zijn om een webpagina te zien. Het is de informatie die de gebruiker kan zien. CSS is de opmaak van die informatie en JS vormt de achterliggende code die de functionaliteit van de website verzorgd. Om een goed design te kunnen maken werkt men via sprints.

Designsprints zijn een intensief proces van vijf dagen tot twee weken waarin teams met een gebruikersgerichte aanpak ontwerpproblemen aanpakken. Het is een manier om uitdagingen in kaart te brengen, oplossingen te verkennen, de beste te kiezen, een prototype te maken en dit te testen. [8] De gebruikerservaring staat hier centraal.

3.2.3 INFRASTRUCTUUR

De software heeft ook een infrastructuur nodig om te kunnen draaien. Een goede infrastructuur is de basis van een goed softwareproject. Als de infrastructuur niet in orde is kan de software nooit optimaal werken. De componenten van IT-infrastructuur bestaan uit onderling afhankelijke elementen. De twee kerngroepen zijn hardware en software. Hardware maakt gebruik van software, zoals een besturingssysteem om te kunnen functioneren. Op dezelfde manier beheert een besturingssysteem systeembronnen en hardware. Besturingssystemen leggen ook verbindingen tussen softwaretoepassingen en fysieke middelen met behulp van netwerkcomponenten. [9] Dit kan ook allemaal virtueel gebeuren door het gebruik van virtuele machines. Hierbij wordt hardware virtueel voorzien binnen een bestaand OS. Hierdoor kan deze virtuele machine fungeren als een echte fysieke machine. Er kunnen meerdere virtuele "gast" machines draaien op een fysieke "host" machine. [10]

3.3 VERGELIJKING

3.3.1 PROGRAMMEERTAAL

Python zou een goede keuze zijn voor het ontwikkelen van een firewall door zijn vele beschikbare modules en bibliotheken. Deze maken het implementeren van netwerkfunctionaliteiten en beveiliging veel makkelijker. Nog een belangrijke reden is dat het hele team met deze taal vertrouwd is.

Verder is bij de bestaande compatibiliteit met iptables door de module netfilterqueue ook zeer belangrijk. Ook hier heeft Python de bovenhand in vergelijking met C#. Python draait bovendien standaard op Linux. Hierdoor kunnen nieuwe functionaliteiten snel worden toegevoegd aan de firewall.

3.3.2 USERINTERFACE

Een groot voordeel van een web userinterface tegenover een applicatie userinterface is dat het niet afhankelijk is van het besturingssysteem van de host. Websites zijn zeer universeel en zo kan de userinterface van overal en op elk apparaat met of zonder een internetverbinding beken worden. De toegankelijkheid is op deze manier gegarandeerd.

3.3.3 INFRASTRUCTUUR

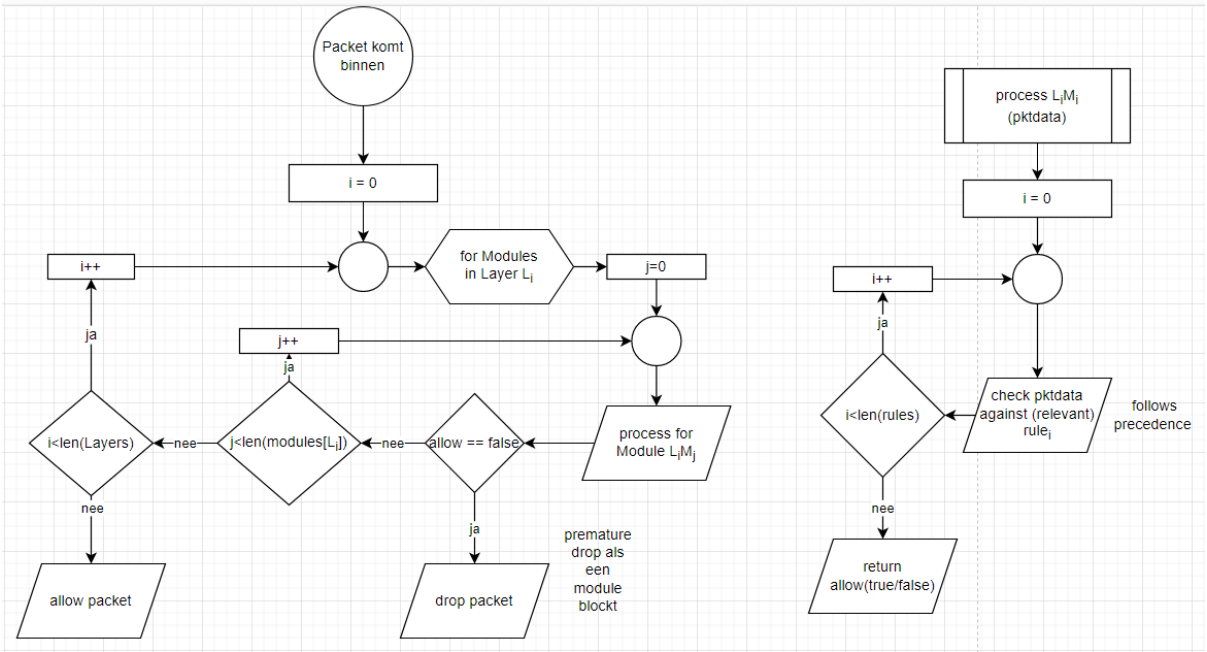
Het gebruik van fysieke of virtuele hardware is afhankelijk van verschillende factoren. Financiële, praktische of bepaalde noden kunnen de keuze in zekere zin beïnvloeden. Toch heeft virtualisatie zekere voordelen. Er kan sneller hersteld worden door het terugkeren naar een eerder opgeslagen werkende configuratie. De snelheid bij het opzetten van een testomgeving. Het gebruik kunnen maken van bestaande malware in een veilige omgeving zonder gevaar op zelfbesmetting.

4 GEKOZEN OPLOSSING

4.1 FIREWALL

Als programmeertaal voor de applicatie werd voor python gekozen. Er werd ook gekozen voor een webapplicatie. De gebruikers kunnen communiceren met de firewall door middel van een webpagina.

De werking van de firewall wordt hieronder geïllustreerd met een block diagram (figuur 1). Wanneer een pakket binnenkomt zal die in een lus terecht komen. Deze lus zal alle modules die tot een bepaalde laag behoren uitvoeren. Binnen een module wordt de relevante data vergeleken met de regels binnen de module. Hier volgt uit of er aan de regels van de module wordt voldaan. Als de regel voldoet, kan de firewall het pakket laten vallen of doorlaten. Belangrijk is dat de data tegenover alle regels binnen een module zal getest worden. Als de data volgens de regels niet geblokkeerd hoeft te worden zal het naar de volgende module gestuurd worden om daar hetzelfde proces te doorlopen. Dit tot alle modules doorlopen zijn. Vervolgens zal naar de volgende laag gekeken worden en begint alles weer van in het begin. Pas wanneer alle lagen doorlopen zijn zal het pakket geaccepteerd worden.



Figuur 1: Block Diagram Firewall

Bij het opstarten van de firewall, moeten alle modules eerst worden ingeladen. Dit wordt gerealiseerd in onderstaande code.

```
#!/usr/bin/python
#example code
import os
import importlib

# Define the path to the modules folder
module_folder = "Backend/modules"

def getModules(moduleset,ignored=[]):
        # Get a list of all Python files in the modules folder
        module_filenames = [f for f in
os.listdir(f"{module_folder}/{moduleset}") if
os.path.isdir(f"{module_folder}/{moduleset}/{f}") and f!="__pycache__"]

        # Import all modules in the modules folder and store them in a
dictionary
        modules = {}
        for module_name in module_filenames:
                try:
                        module =
importlib.import_module(f"{module_folder.replace('/','.')}.{moduleset}.{module_n
ame}",package=None)
                        modules[module_name] = (module, (module_name in
ignored))
                except Exception as ex:
                        print(f"error loading module
{module_name}.{moduleset}")
                        print(ex)
        return modules
```

Codefragment 1: modules inladen

Een volgend belangrijk deel in de code is de firewallChannel functie. Die wordt gedefinieerd als een hogere-ordefunctie die een specifiek verwerkingskanaal voor de firewall retourneert op basis van de opgegeven richting ("in", "out" of "fwd"). Hieronder wordt de implementatie getoond (codefragment 2).

```
def firewallChannel(direction):
    def channel(pkt):
        try:
            sca = IP(pkt.get_payload())#scapy.layers.inet
            logger(direction, sca.src, sca.dst, sca.name, sca.payload.name,
sca.payload.payload.name)
            #
            for module in config_object[0]["L3_modules"]: #for key in dictionary
                if not config_object[0]["L3_modules"][module][1]:
                    #guard clause, module is marked disabled
                    continue
                direct = ipparser.getDirection(pkt.dst, INTERNAL, INTERNAL_MASK)
                accepted = config_object[0]["L3_modules"][module][0].run(direct,
sca)
                if(not accepted):
                    pkt.drop()
                    break
            pkt.accept()
        except Exception as e:
            print(e)

    return channel
```

Codefragment 2: firewallChannel functie

Tot slot volgt een deel van de firewall rules module (codefragment 3). Deze is werkzaam op laag 3 en voert een SQL-query uit op de firewallrules-tabel in de database om te controleren of het gegeven pakket moet worden toegestaan of geblokkeerd op basis van de firewallregels.

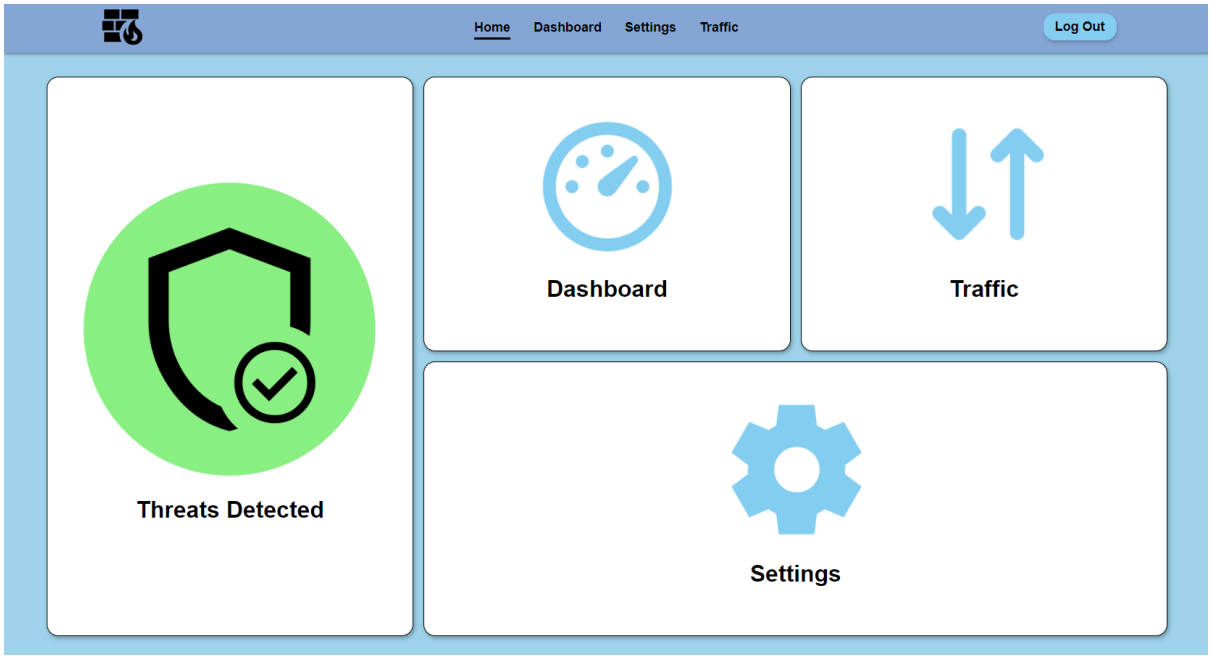
```
def run(direction="ingress",pkt=None):
    #query="WITH filteredTable AS (SELECT * FROM firewallrules WHERE direction=?
and src_ip=?&src_mask and dst_ip=?&dst_mask and (protocol=? or protocol is null)
and port_start<=? and port_end>=?) SELECT * FROM filteredTable WHERE
priority=(SELECT MIN(priority) FROM filteredTable)"
    query="WITH filteredTable AS (SELECT * FROM firewallrules WHERE
src_ip=?&src_mask and dst_ip=?&dst_mask and (protocol=? or protocol is null) and
port_start<=? and port_end>=?) SELECT * FROM filteredTable WHERE
priority=(SELECT MIN(priority) FROM filteredTable)"
    ip_src = pkt.src
    ip_dst = pkt.dst
    protocol = pkt.get_field('proto').i2s[pkt.proto].upper()
    try:
        port=pkt.payload.dport
    except Exception as ex:
        port = 0 #gebruik port 0 voor packets zonder poort

    queryparams = (
        #direction,
        ipparser.parse_to_int(ip_src),
        ipparser.parse_to_int(ip_dst),
        protocol,
        port, port
    ) #port x2 omdat port range test op zelfde cijfer test
    try:
        matchedrules = con().execute(query, queryparams)
        for rule in matchedrules:
            if(rule[2]==0):
                return False
        return True
    except Exception as ex:
        print("exception during sql operation\n\t" + str(ex))
```

Codefragment 3: firewall rules run functie

4.2 USERINTERFACE

De userinterface wordt zodanig ontwikkeld dat deze toegankelijk is voor iedereen. Op de homepagina (figuur 2) wordt gekozen voor grote duidelijke knoppen met de meest frequente en belangrijkste gebruikte functionaliteiten. De status van de firewall wordt hier ook getoond. Het geeft aan of er dreigingen gevonden werd en er pakketten werden geblokkeerd. Wanneer er gekozen wordt om hierop te drukken, dan gaat de gebruiker naar de dashboard pagina. Deze kan ook bereikt worden via de dashboard knop zelf. Hier wordt een gedetailleerde weergave van de meldingen getoond. Wanneer men het netwerkverkeer wil bekijken, kan men kiezen om op de traffic knop te drukken. Als de instellingen moeten worden aangepast kan er gekozen worden om op de instellingen knop te drukken.



Figuur 2: Homepagina

Om de gebruiksvriendelijkheid te verbeteren dienen er testen te worden afgenomen. Deze zijn niet afgenomen bij een test publiek omdat de focus van dit project gericht is naar zijn functionaliteit.

4.3 INFRASTRUCTUUR

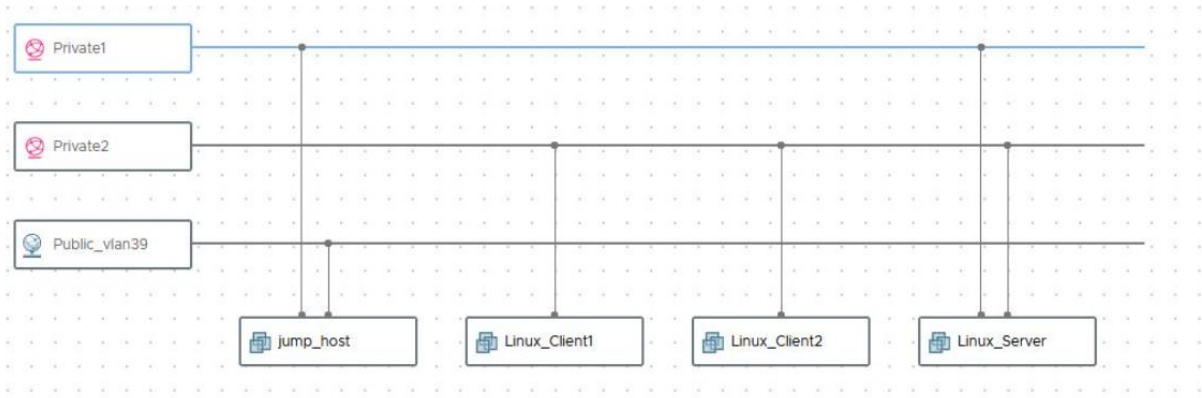
4.3.1 VIRTUELE MACHINES

De software wordt voorzien op een CentOS 8 virtuele machine. Alle virtuele machines worden in dit een gesloten netwerk geplaatst waar de nodige configuraties worden op toegepast. Dit wordt hieronder besproken. Volgende besturingssystemen worden gekozen:

- CentOS 8 voor de firewall;
- Fedora voor de clients;
- Windows 10 voor de jump host.

4.3.2 NETWERK

Voor het project wordt er gekozen voor een gesloten omgeving, zodat er geen gevaar ontstaat voor zelfbesmetting bij het testen met bestaande malware. In de figuur hieronder (figuur 3) wordt de opstelling afgebeeld. De virtuele machines met als prefix “Linux” behoren tot de testomgeving. De “jump host” is nodig om via SSH (secure shell) verbinding te maken met de firewall server. Het verkeer van de twee computers in het netwerk wordt gerouteerd door de firewall server zelf. Zo werkt deze ook als een eenvoudige router. Alle toestellen binnen dit afgesloten netwerk zijn via SSH-gateway bereikbaar vanop de jump host.



Figuur 3: Netwerk Configuratie

5 EVALUATIE

Voor de evaluatie van dit project, kan er geconstateerd worden dat er een goede samenwerking tussen de teamleden was. Het gebruik van Gitlab heeft zeker bijgedragen aan de ontwikkeling van het project. Zo werden de sprints goed verdeeld en hadden ze een haalbaar gewicht, waardoor deadlines werden behaald. Dit platform is zeer goed voor zowel het plannen als het uitvoeren van grote projecten.

Door de complexiteit werden niet alle bedachte modules ontwikkeld. Enkel het opvragen van data en het toevoegen van firewall regels is geïmplementeerd. Doordat de opbouw voorzien is in modules, kunnen nieuwe modules eenvoudig worden geïmplementeerd.

Het software gedeelte diende vijf weken voor de deadline van dit rapport klaar te zijn. Ook dit werd behaald, waardoor de focus verder kon gelegd worden op het schrijven van dit rapport.

Tijdens het ontwikkelen van de software werden enkele uitdagingen overwonnen. Zo is er een sterke focus op security en wordt er gebruik gemaakt van tokens. Die worden steeds gevalideerd in de request data. Omdat de website niet rechtstreeks gekoppeld is aan de firewall en door de modulaire opbouw van het systeem, heeft het inwerken van de webrequest meer tijd nodig gehad dan verwacht.

Een andere factor die een invloed had op het process was het gebrek aan documentatie over installeren van de netfilter_queue-devel rpm package op Rocky Linux 9. De naam van van de Git Repository is namelijk veranderd van powertools naar crb. Veel documentatie verwijst nog naar de ondertussen niet bestaande powertools packages. Hierdoor heeft het installeren van c++ header files veel tijd in beslag genomen.

CONCLUSIE

Het doel van het project is om een educatieve bijdrage te leveren aan de opleiding Bachelor Elektronica-ICT. Het brengt inzicht in de werking van een firewall. Het project met zijn onderzoek naar malware en het ontwikkelen van de firewall, bieden een meerwaarde in de algemene beroepskennis.

Het resultaat is een werkende firewall geschreven in python en met een werkende nette interface. Verschillende modules kunnen in- en uitgeschakeld worden zodat de gebruiker dit volgens zijn noden kan aanpassen. Verder kan een gebruiker ook zijn eigen specifieke regels toevoegen.

Het project heeft veel ruimte voor uitbreiding. Zo wordt nu bij het bestuderen van de data enkel naar laag drie en vier van het OSI-model gekeken. Dit staat hardcoded in de module en zo is daar ruimte voor uitbreiding naar andere lagen.

Er zijn op vlak van de modules zelf ook wel nog wat uitbreidingen mogelijk. Zo kunnen bijvoorbeeld modules toegevoegd worden. Verder moeten dan ook de webrequests verder uitgewerkt worden. Ideaal zou dan zijn dat er ook aan het inlogsysteem gewerkt wordt om met verschillende gebruikers te kunnen werken. Het toevoegen van een databank om alle gegevens van de verschillende gebruikers te kunnen opslaan is hierbij wel een vereiste.

Ten slotte zou ook het originele idee waaruit dit project gegroeid is kunnen toegepast worden. Het implementeren van artificiële intelligentie (AI) op dit project is een project op zichzelf. Het zoeken naar correcte data om een model te trainen die malware kan onderscheiden van onschadelijke software zou enorm veel tijd kosten. Bij dit model moet ook gezorgd worden dat het accuraat genoeg is zodat het zeker niet storend wordt voor de gebruiker wanneer steeds onschadelijke pakketten geblokkeerd worden. Nog een belangrijk punt is dat de extra tijd die nodig is om een pakket met AI te analyseren zo laag mogelijk gehouden wordt.

HANDLEIDING

Het commando sudo is vereist om volgende opdrachten met beheerdersrechten uit te voeren.

Het opzetten van iptables:

1 Open een terminal of opdrachtprompt.

2 Voer de volgende opdracht uit om iptables-regels in te stellen en alle inkomende pakketten door te sturen naar nfqueue 1, 2 en 3 voor respectievelijk de INPUT, OUTPUT en FORWARD chains:

```
$ sudo ‘code/backend/startup scripts/setiptables’
```

3 Het programma starten:

```
$ sudo python code/main.py
```

4 Betekenis van de parameters:

- s website starten

- f firewall starten

5 Het programma stoppen:

Om het programma op een goede manier te stoppen, druk je op Ctrl+C in de terminal waarin het programma wordt uitgevoerd.

6 Indien er nog processen niet worden afgesloten kan gekozen worden voor:

```
$ sudo ‘code/backend/startup scripts/killall’
```

Opmerking: Bij het uitvoeren van het killall-commando voor de firewall.py implementatie kan de terminal zich vreemd gedragen.

7 Herstel iptables naar de standaardinstellingen door het volgende commando uit te voeren:

```
$ sudo iptables -F
```

LITERATUURLIJST

[1] L. M. Pelzer, „The True Cost of Cybersecurity Incidents,” paloalto networks, 06 2021. [Online]. Available: https://www.paloaltonetworks.com/blog/2021/06/the-cost-of-cybersecurity-incidents-the-problem/. [Geopend 20 03 2023].

[2] Check Point, „Key Ingredients of a Strong Firewall,” Check Point, 2023. [Online]. Available: https://www.checkpoint.com/cyber-hub/network-security/what-is-firewall/key-ingredients-of-a-strong-firewall/. [Geopend 20 03 2023].

[3] Check Point, „The Different Types of Firewalls,” Check Point, 2023. [Online]. Available: https://www.checkpoint.com/cyber-hub/network-security/what-is-firewall/the-different-types-of-firewalls/. [Geopend 20 03 2023].

[4] Fortinet, „Stateful & Stateless Firewall Differences,” Fortinet, 2023. [Online]. Available: https://www.fortinet.com/resources/cyberglossary/stateful-vs-stateless-firewall. [Geopend 21 03 2023].

[5] Cisco, “What is a next-generation firewall?,” Cisco, 2023. [Online]. Available: https://www.cisco.com/c/en/us/products/security/firewalls/what-is-a-next-generation-firewall.html. [Accessed 22 03 2023].

[6] GeeksforGeeks, "Python Language advantages and applications," GeeksforGeeks, 24 April 2023. [Online]. Available: https://www.geeksforgeeks.org/python-language-advantages-applications/. [Accessed 12 Mei 2023].

[7] R. Payne, “Advantages of C#,” codeguru, 23 Januari 2023. [Online]. Available: https://www.codeguru.com/csharp/c-sharp-advantages/. [Accessed 12 Mei 2023].

[8] Interaction Design Foundation, “Design Sprints,” Interaction Design Foundation, [Online]. Available: https://www.interaction-design.org/literature/topics/design-sprints. [Accessed 20 Mei 2023].

[9] IBM, “What is IT Infrastructure?,” [Online]. Available: https://www.ibm.com/topics/infrastructure. [Accessed 20 Mei 2023].

[10] vmware, “Virtual Machine,” vmware, [Online]. Available: https://www.vmware.com/topics/glossary/content/virtual-machine.html#:~:text=A%20Virtual%20Machine%20(VM)%20is,a%20physical%20%E2%80%9Chost%E2%80%9D%20machine. [Accessed 20 Mei 2023].

[11] Fortinet, „How does a firewall work?," Fortinet, 2023. [Online]. Available: https://www.fortinet.com/resources/cyberglossary/how-does-a-firewall-work. [Geopend 20 03 2023].

[12] Fortinet, „Hybrid Firewall Advantages and Disadvantages," Fortinet, 2023. [Online]. Available: https://www.fortinet.com/resources/cyberglossary/hybrid-firewall-advantages-disadvantages. [Geopend 20 03 2023].

[13] Cisco, „What is a firewall?," Cisco, 2023. [Online]. Available: https://www.cisco.com/c/en/us/products/security/firewalls/what-is-a-firewall.html. [Geopend 20 03 2023].

[14] Sangfor Technologies, „What is Software Firewall? Difference between Hardware Firewall and Software Firewall," Sangfor, 2023. [Online]. Available: https://www.sangfor.com/blog/cybersecurity/what-software-firewall-difference-between-hardware-firewall-and-software. [Geopend 21 03 2023].

BIJLAGEN

BIJLAGE 1: KOPIEËN VAN DATASHEETS

BIJLAGE 2: VERGADERVERSLAGEN

Vergadering zondag 5 februari 2023:

- Bespreking aanpak project;
- Opzoeken informatie rond malware;
- Planning en taakverdeling.

Vergadering zondag 19 februari 2023:

- Tussentijdse code evaluatie;
- Blokdiagram dient opnieuw te worden ontworpen;
- Basis documentatie in Word document voorzien;
- Planning en taakverdeling.

Vergadering zondag 19 maart 2023:

- Tussentijdse code evaluatie;
- Userinterface eerste demo;
- Bespreking van het rapport;
- Planning en taakverdeling.

Vergadering zondag 23 april 2023:

- Demonstratie code;
- Userinterface tweede demo;
- Bespreking van het rapport;
- Planning en taakverdeling.

Vergadering vrijdag 5 mei 2023:

- Demonstratie project;
- Bespreking van het rapport;
- Planning en taakverdeling.

Vergadering zondag 28 mei 2023:

- Aanpassingen maken aan eindrapport
- Goedkeuring voor indiening door het team.

BIJLAGE 3: LOGBOEK RAPPORTEREN

Jonas Van den Berghe	N.v.t.	Schrijven inleiding, abstract, toevoegen titels, maken van inhoudsopgave.
Vasseur Michaël	Kaft, Abstract, 8	Kaft voorzien van figuur met titel en subtitel. Abstract en inleiding herschreven.
Kenzo Staelens	1	Inhoudsopgave genereren en paginanummers toevoegen.
Jonathan van Caloen	N.v.t, Kaft, 9-12	Aanpassen van titels en subtitels van de hoofdstukken. Herwerken structuur kaft, herwerken structuur document, H1 firewalls geschreven, bronnenlijst aangemaakt
Jonas Van den Berghe	9-12	Aanpassen H1 firewalls
Kenzo Staelens	9-12	Aanpassen H1 firewalls
Vasseur Michaël	9-12	Aanpassen H1 firewalls
Jonas Van den Berghe	9-12	Verbeteren H1 firewalls
Vasseur Michaël	9-12	Verbeteren H1 firewalls
Jonas Van den Berghe	13-17	Schrijven H2 malware
Vasseur Michaël	13-17	Herschrijven H2 malware
Kenzo Staelens	13-17	Verbeteren H2 malware
Jonas Van den Berghe	final	Invulling eind onderdelen
Kenzo Staelens	final	Verbeteren eind onderdelen
Vasseur Michaël	final	Invulling/ Verbeteren eind onderdelen
Jonathan van Caloen	final	Verbeteren eind onderdelen