Universidade de Aveiro
2018

Departamento de Eletrónica,
Telecomunicações e Informática

João Manuel Ribeiro Mota

Detecção de Anomalias na Partilha de Ficheiros em Ambientes Empresariais

File Sharing Anomaly Detection in Business Environments

Universidade de Aveiro
2018

Departamento de Eletrónica,
Telecomunicações e Informática

João Manuel Ribeiro Mota

Detecção de Anomalias na Partilha de Ficheiros em Ambientes Empresariais

File Sharing Anomaly Detection in Business Environments

Tese apresentada à Universidade de Aveiro para cumprimento dos requisitos necessários à obtenção do grau de Mestre em Engenharia de Computadores e Telemática, realizada sob a orientação científica do Doutor António Manuel Duarte Nogueira, Professor Auxiliar do Departamento de Eletrónica, Telecomunicações e Informática da Universidade de Aveiro, e do Doutor Paulo Jorge Salvador Serra Ferreira, Professor Auxiliar do Departamento de Eletrónica, Telecomunicaões e Informática da Universidade de Aveiro
.

o júri / the jury

presidente / president

Professor Doutor Tomás Oliveira e Silva
Professor Associado da Universidade de Aveiro

vogais / examiners committee

Professor Doutor Joel José Puga Coelho Rodrigues
Professor do Instituto Nacional de Telecomunicações (Inatel), Brasil e Professor Auxiliar Convidado com Agregação do Departamento de Informática da Faculdade de Engenharia da Universidade da Beira Interior

Professor Doutor António Manuel Duarte Nogueira
Professor Auxiliar da Universidade de Aveiro

agradecimentos /
acknowledgements

Aproveito para agradecer ao orientador Professor Doutor António Manuel Duarte Nogueira, pela sua disponibilidade, simpatia, por todo o conhecimento que me passou, e por me guiar sempre na direção certa.
Ao professor Doutor Paulo Jorge Salvador Serra Ferreira, pela sua disponibilidade e pelas sugestões pertinentes.
Aos meus pais e irmãos, pelo apoio incondicional prestado estes anos e por nunca deixarem de acreditar em mim.
Aos meus amigos, familiares e colegas por todo apoio.
Por fim, à minha namorada Sara por todo o apoio, paciência e também por nunca deixar de acreditar em mim.
Obrigado!

Palavras Chave

redes, detecção de anomalias, partilha de ficheiros, ransomware, roubo de dados, propagação, machine learning

Resumo

Partilha de ficheiros é a atividade de disponibilizar ficheiros (documentos, vídeos, fotos) a utilizadores. As empresas usam a partilha de ficheiros para disponibilizar ficheiros aos seus utilizadores e trabalhadores. A disponibilidade destes ficheiros pode ser feita a partir de uma rede interna, serviço de nuvem (externo) ou até Ponto-a-Ponto. Normalmente, os ficheiros contidos no serviço de partilha de ficheiros contêm dados confidenciais que não podem ser divulgados. O ataque de violação de dados realizado a Equifax explorou uma vulnerabilidade de dia zero que permitiu execução de código arbitrário, levando a que a informação de 143 milhões de utilizadores fosse comprometida. Ransomware é um tipo de malware que cifra os dados do computador (documentos, multimédia...) tornando-os inacessíveis ao utilizador, exigindo a este um resgate para decifrar esses dados. Este tipo de malware tem sido uma grande ameaça às empresas atuais. WannaCry e NotPetya são alguns exemplos de Ransomware que tiveram um grande impacto com grandes quantias de resgate, WannaCry alcançou mais de 142,361.51$ em resgates. Neste tabalho, propomos um sistema que consiga detectar anomalias na partilha de ficheiros, como o ransomware (WannaCry, NotPetya) e roubo de dados (violação de dados Equifax), bem como a sua propagação. A solução consiste na monitorização da rede da empresa, na criação de perfis para cada utilizador/máquina, num algoritmo de machine learning para análise dos dados e num mecanismo que bloqueie a máquina afetada no caso de se detectar uma anomalia.

Keywords network, anomaly detection, file sharing, ransomware, theft, propagation, machine learning.

Abstract File sharing is the activity of making archives (documents, videos, photos) available to other users. Enterprises use file sharing to make archives available to their employees or clients. The availability of these files can be done through an internal network, cloud service (external) or even Peer-to-Peer (P2P). Most of the time, the files within the file sharing service have sensitive information that cannot be disclosed. Equifax data breach attack exploited a zero-day attack that allowed arbitrary code execution, leading to a huge data breach as over 143 million user information was presumed compromised. Ransomware is a type of malware that encrypts computer data (documents, media, ...) making it inaccessible to the user, demanding a ransom for the decryption of the data. This type of malware has been a serious threat to enterprises. WannaCry and NotPetya are some examples of ransomware that had a huge impact on enterprises with big amounts of ransoms, for example WannaCry reached more than 142,361.51$ in ransoms. In this dissertation, we purpose a system that can detect file sharing anomalies like ransomware (WannaCry, NotPetya) and theft (Equifax breach), and also their propagation. The solution consists of network monitoring, the creation of communication profiles for each user/machine, an analysis algorithm using machine learning and a countermeasure mechanism in case an anomaly is detected.

Contents

List of Figures

List of Tables

Glossary

BTC	Bitcoin
P2P	Peer-To-Peer
OSI	Open Systems Interconnection
ARP	Address Resolution Protocol
IP	Internet Protocol
TCP	Transport Control Protocol
UDP	User Datagram Protocol
ICMP	Internet Control Message Protocol
SMB	Server Message Block
SMBv1	Server Message Block version 1
LAN	Local Area Network
CIFS	Common Internet File System
VLAN	Virtual Local Area Network
HTTP	Hypertext Transfer Protocol
HTTPS	Hypertext Transfer Protocol Secure
SSL	Secure Socket Layer
TLS	Transport Layer Security
TFTP	Trivial File Transfer Protocol
FTP	File Transfer Protocol
SFTP	Secure File Transfer Protocol
SSH	Secure Shell
WebDAV	Web Distributed Authoring and Versioning
WebDAVs	SSL Web Distributed Authoring and Versioning
AFTP	Accelerated File Transfer Protocol
AS2	Applicability Statement 2
EDI	Electronic Data Interchange
VPN	Virtual Private Network
IDS	Intrusion Detection System
IPS	Intrusion Prevention System
DoS	Denial of Service
SQL	Structured Query Language
NSA	National Security Agency
FEA	File Extended Attributes
OS2	Operating System 2
TAP	Terminal Access Point
RMON	Remote Network MONitoring
RUM	Real User Monitoring
ML	Machine Learning
AI	Artificial Intelligence
OCR	Optical Character Recognition
SVM	Support Vector Machines
PCA	Principal Component Analysis
TP	True Positives
FN	False Negatives
TN	True Negatives
FP	False Positives
ID	Intrusion Detection
IP	Intrusion Prevention
NIDS	Network-based Intrusion Detection System
HIDS	Host-based Intrusion Detection System
NIPS	Network-based Intrusion Prevention System
HIPS	Host-based Intrusion Prevention System
DIDPS	Distributed Intrusion Detection and Prevention System
IDPS	Intrusion Detection and Prevention System
ANIDS	Anomaly-based network Intrusion Detection System
ANN	Artificial Neural Networks
GA	Generic Algorithm
PSO	Particle Swarm Optimization
ICPSO	Improved Particle Swarm Optimization
ADTree	Alternating Decision Tree
CUSUM	Cummulative Sum
PAYL	Payload-based anomaly detector
N@G	Network at Guard
STAT	STATistical anomaly detection
FSAS	Flow-based Statistical Aggregation Schemes

WVR	Weight Voting Rule
CM	Confusion Matrix
ROC	Receiver Operating Characteristics
ADMIT	Anomaly-based Data Mining for InTrusions
LCS	Longest Common Subsequence
DIDS	Distributed Intrusion Detection System
ODC	Outlier Discovery and Clustering
SSE	Sum if Squared Error
SST	Total Sum of Squares
SNORT	SNORT
PE	Processing Elements
HIDE	Hierarchical Network Intrusion Detection System
IDA	Intrusion Detection Agent
OS	Operating System
AES	Advanced Encryption Standard
SAML	Security Assertion Markup Language
VM	Virtual Machine
RBF	Radial Basis Function
DT	Decision Tree
GB	Gradient Boosting
GBM	Gradient Boosting Machine
SVC	C-Support Vector Classification

CHAPTER 1

Introduction

1.1 Motivation

Enterprises waste more and more money on security, in 2018 money spent in security is going to reach 96 billions of dollars, which is 8% more than in the year 2017 [1]. Cyber attacks such as WannaCry and NotPetya, and also Equifax breach were large-scale attacks with high amounts of ransom which served as an open-eye for enterprises and the importance of security. WannaCry, and NotPetya, are ransomwares that encrypt all computer data and ask for a ransom (example: 300 dollars) to unencrypt the data. The payment is usually done with virtual currency like Bitcoin [2]. WannaCry only, has reached a total sum of ransoms of 142,361.51$ [3] in bitcoins (52.1966 Bitcoin (BTC), the total value in dollars can change with BTC market price).

Enterprise networks are constantly being attacked and most of the attacks are able to penetrate enterprise traditional defenses like firewall and anti-virus. Most of the attacks are detected by these defenses, although there are a few that are not known and consequently, not detected. Those undetected attacks are called zero-day. Zero-day are system vulnerabilities that are not yet known to people that would like to mitigate the vulnerability and can be used as an entry point in a vulnerable system.

The source of the attacks can be either internal or external. External attacks are normally detected by current defenses. Where internal attacks like a pen inputted in a computer or a virus that is executed locally, are difficult to detect and can be a problem. One of the biggest problems of traditional security systems is the detection of virus propagation when the machine is already infected. There is an urgent need of reinforcing current security systems that can detect these attacks and limit their propagation. The goal of these attacks is mostly theft of sensitive information, denial of service, or ransom. File sharing is the activity of making archives (documents, videos, photos) available to other users. The availability

[1] `https://www.gartner.com/newsroom/id/3836563`, last access 24/07/2018
[2] `https://pt.wikipedia.org/wiki/Bitcoin`
[3] `https://www.thesslstore.com/blog/wannacry-ransom-total/`, last access 23/07/2018

of these archives can be done through an internal network, external cloud server or even Peer-To-Peer (P2P) where files are stored in a user computer. File sharing in a business environment is very common nowadays. People within the enterprise have the need to share files between employees or even clients, and most of the time, the files being shared have sensitive information that shouldn't be disclosed.

Intrusion Detection and Prevention Systems (IDPSs) came to help traditional defenses protecting the network. They came as a second layer of network security which monitors the network and analysis all data in order to detect malicious activities or policy violations. This kind of systems are normally used for general network defense and not specifically for file sharing anomaly detection. They are still in development as there is the urge of decreasing false positive rates.

The detection of enterprise security threats propagation in file sharing in an enterprise network requires network monitoring, the creation of communication profiles for each machine/user of the network and the detection of outlier behaviors.

The objective of this study is the detection of file sharing anomalies such as ransomware, theft, as well as their propagation. The detection is performed by monitoring the network and using our trained machine learning approach to classify data as normal and abnormal. To train the machine learning we first have to create both normal and abnormal network profiles.

1.2 Contribution

This dissertation was developed at IT - Institute of Telecommunications Aveiro.

As mentioned in 1.1 enterprise networks are constantly targetted with attacks that are not detected by traditional security defenses, as there is a need of developing a new security module that helps in the detection of those threats, isolate the affected machine and limit its propagation.

As said in section 1.1, the goal of this study is the detection of file sharing anomalies. In order to achieve that goal, we have to gather network normal and abnormal information, train the machine learning model with both normal and abnormal data, and test the model with 20% of the training data set. For better model evaluation we also tested the machine learning model with a new dataset containing unseen data. Since we know specifically the anomalies that we want to detect and we can reproduce them, we used a machine learning classification approach. For the profile's creation, we transformed network packets into observation windows where we counted our packet relevant features and applied statistics to those countings. We also used a sliding window approach as it offers more observations in an observation window. We achieved quite good results with our approach, which are shown in chapter 5.

Our solution has the limitation of not detecting outside or locally executed attacks. The goal is the detection of internal attacks, their propagation and the use of countermeasures to limit their propagation. Considering that the machine that we are analyzing is already infected, our solution is to study the profile of the machine, compare all new communications made by this machine with its profile and check if they are anomalous activities. If anomaly

communications are detected, countermeasures such as limiting or blocking the machine are used.

1.3 Dissertation Structure

This dissertation is divided into seven chapters. Chapter one has a brief introduction to the problem, the objective of this study and the main results that were achieved. Chapter two consists of all theoretic information needed in order to understand the rest of the document. In chapter three we present several solutions that were proposed by different authors for the detection of network anomalies using machine learning approaches. Chapter four describes how we developed our solution. Chapter five presents of our experiment results obtained by using different machine learning approaches. Chapter six presents the main conclusions and some topics for future research. Finally, in chapter seven we include the rest of the experiments table results.

CHAPTER 2

Theoretical Context

2.1 Network Communication

In general, communication is defined as the act of exchanging information from one entity or group of entities to another. In communication networks, this definition is also applied. Exchanged information is identified as data and the entities or groups of entities are identified as endpoints of the communication. In communication, a set of rules is needed in order to understand exchanged information. In communication networks, we call those rules a protocol. So, a protocol specifies the set of rules that are needed to communicate. Generally, data transmission is done by sending messages, which can be divided into small blocks called packets.

A packet is a formatted unit of data containing bits or bytes and is carried by a packet-switched network. A packet content can be divided into two parts: control information and user data. Control information, usually found in packet headers and trailers, provides all information that is needed to accomplish the correct delivery of the actual data. Control information includes fields such as source and destination addresses, error detection codes, and sequencing information.

Control information ensures that data is sent to the correct endpoints and protocol guarantees that both endpoints are following the same set of rules. Since the protocols main function is end-to-end communication, they can be inserted into the transport layer in both Transport Control Protocol (TCP)/Internet Protocol (IP) and Open Systems Interconnection (OSI) models. This type of communication can be generally divided into connection-oriented and connectionless communication. The Connection-oriented approach is implemented by the TCP, while the connectionless approach is implemented by the User Datagram Protocol (UDP). The Internet Control Message Protocol (ICMP) is used for control purposes.

The OSI model is known as the seven-layer model for data communication. The layers are the physical, data link, network, transport, session, presentation and application layers.

The TCP/IP model uses TCP and IP. This model is divided into network access, Internet, transport and application layers.

The main difference between both models is shown in Figure 2.1.

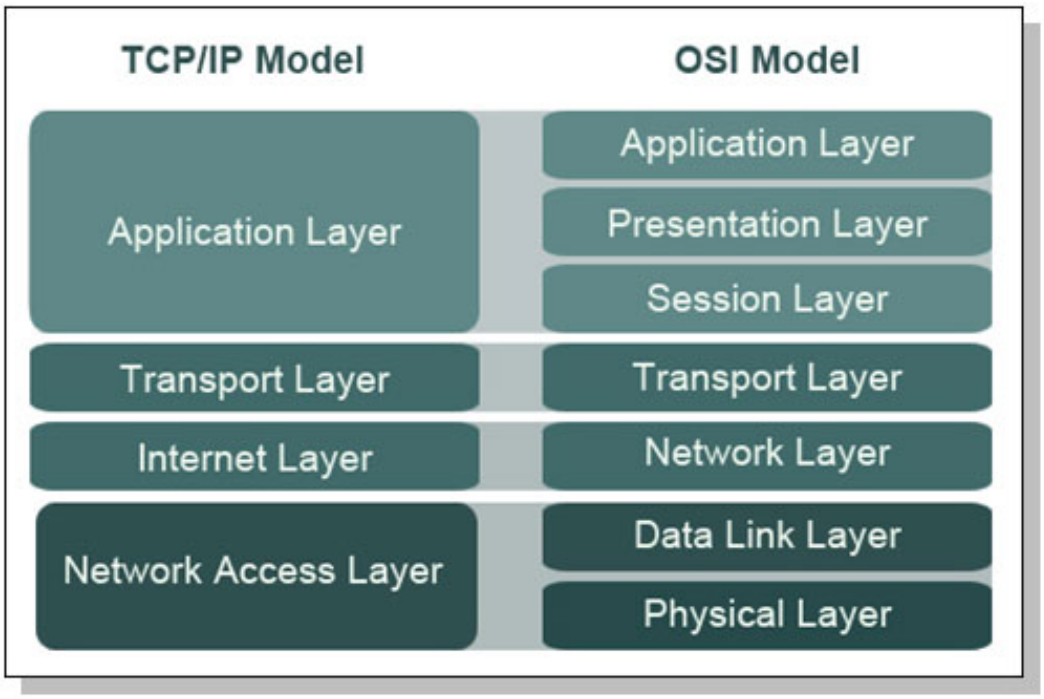

Figure 2.1: OSI Model vs TCP/IP Model [1]

2.1.1 TCP

TCP is a connection-oriented, end-to-end reliable protocol, which provides reliable inter-process communication. TCP provides an abstraction for the communication service between application level and other underlying levels. Data sent within a TCP connection is divided in segments, which are sent as Internet datagrams. The IP header carries several information fields, such as the source and destination ports, the Sequence Number and the Acknowledgement Number. More information about header fields will be shown in Figure 2.2.

[1] Image available at `https://i.stack.imgur.com/FOfAU.jpg`, accessed in 29/10/2018

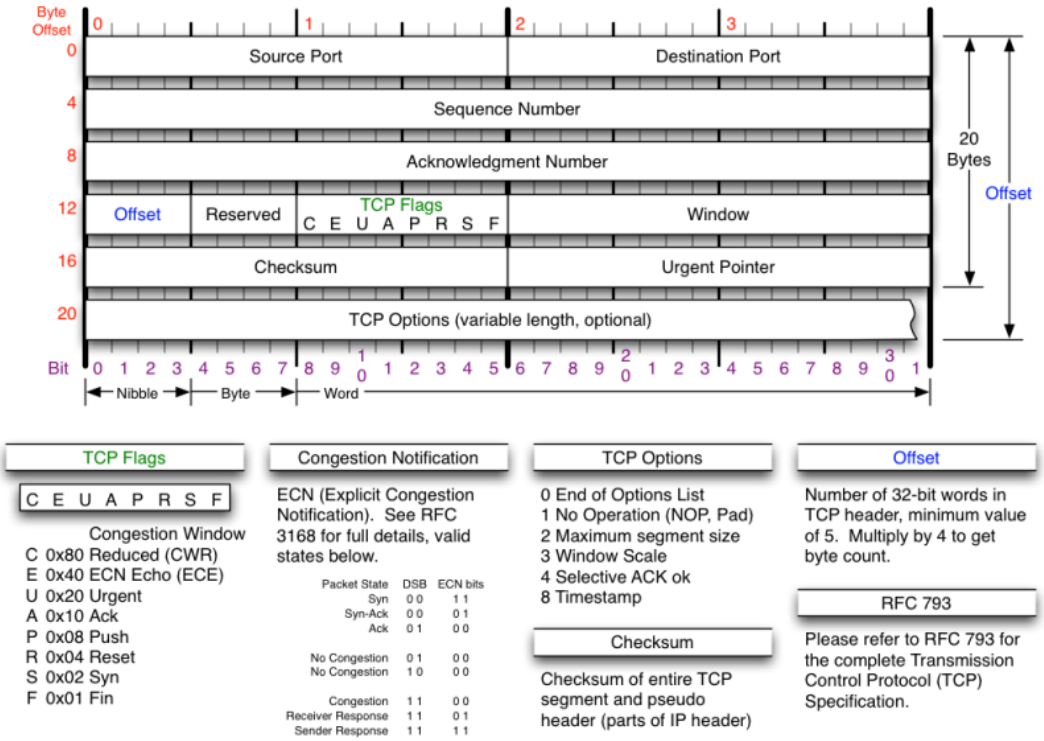

Figure 2.2: Different TCP header fields [2]

2.1.2 UDP

UDP is a message-oriented transport layer protocol that uses IP as the underlying protocol. Messages correspond to datagrams that are sent in the form of packets. UDP does not provide any guarantees to the upper layer protocol about message delivery and it retains no state about the messages that are sent.
UDP fields are shown in Figure 2.3.

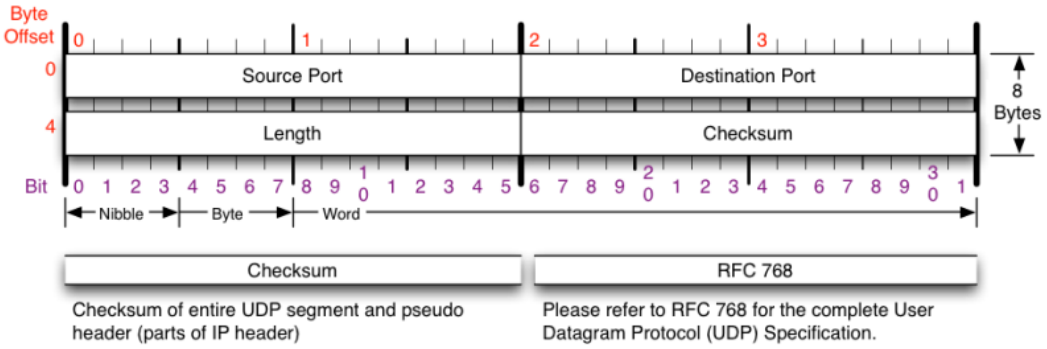

Figure 2.3: Different UDP header fields [3]

[2] Image available: `https://nmap.org/book/images/hdr/MJB-TCP-Header-800x564.png`, accessed in 29/10/2018
[3] Image available: `http://nmap.org/book/images/hdr/MJB-UDP-Header-800x264.png`, accessed in 29/10/2018

2.1.3 ICMP

ICMP is a support protocol for Internet communication. Typically, ICMP messages are used for diagnostic and control purposes or in response to errors that occur in IP packet processing. Error messages are then redirected to the source IP address of the originating packet. ICMP is considered to be an integral part of IP as messages are contained within standard IP packets. ICMP header structure is shown in Figure 2.4.

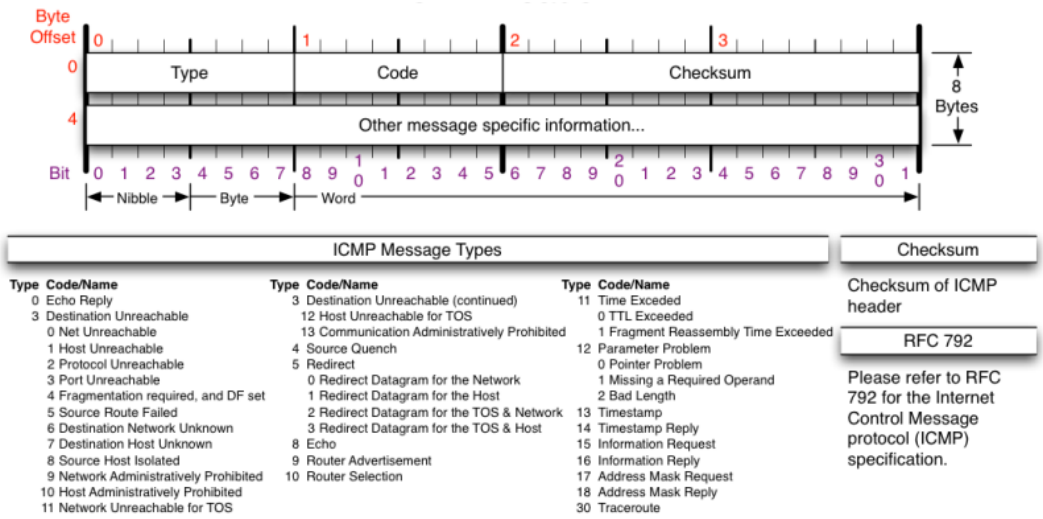

Figure 2.4: ICMP header structure [4]

2.1.4 SMB

Server Message Block (SMB) is part of the application-layer network and provides shared access to files, printers, and serial ports and also communications between nodes on a network. This protocol also provides an inter-process communication mechanism. SMB is mostly used in computers running Microsoft Windows as it is used in Microsoft Windows Network. SMB is also used in Windows services such as Local Area Network (LAN) Manager Server and LAN Manager Workstation. Common Internet File System (CIFS) was an early implementation of SMB also known as Server Message Block version 1 (SMBv1).

2.2 Data link layer

A LAN is a computer network that interconnects computers within a limited area, such as an office, school or university, for data transfer. Computers connected to the LAN are all on the same physical segment.

A Virtual Local Area Network (VLAN) is a simulation of a standard LAN, based on a logical connection, that allows data to be transferred without traditional physical constraints placed on the network. This means that the network administrator can group users into the same VLAN in order to communicate as if they were attached to the same physical segment

[4]Image available: `https://nmap.org/book/images/hdr/MJB-UDP-Header-800x264.png`, accessed in 15/11/2018

when in reality they are in different physical segments. VLAN were developed to offer changes on the network, mainly on how it is managed and administrated. This is possible because VLAN are logical connections, which makes them flexible and flexibility gives more space to network changes.

VLAN are controlled by network switches, while inter-VLAN communication is handled by network routers. Switches are able to logically group users and ports across the network. VLAN are normally used in large flat networks to sub-divide the network into subnets so broadcasting is only made for the intended group and not to all ports. VLAN are widely used in business and campus networks in order to restrict access to certain network places. Groups can have different configurations and security measures. To ensure that a user associated to a group keeps the same VLAN on all campus, an end-to-end VLAN is used, which is a single VLAN associated with switch ports throughout all the campus. These end-to-end VLAN carry the traffic from all VLAN into any configured switch.

2.3 File sharing services and protocols

Enterprises need a way to share files (documents, photos) between their employees or any external client. One good example is a project that is being done by a group of people, so in order for the project to be available for all the employees, a file sharing service can be used. The service can be either hosted on enterprise facilities or on a cloud.
There are several network services and protocols that are used within those services that will be explained briefly below.

Hypertext Transfer Protocol (HTTP) is probably the most known since it defines the interactions between "Web browsers" and "Web servers" and also the formats for the messages used. HTTP uses two kinds of TCP connections: non-persistent, where the client establishes a TCP connection to send a request and the server closes the connection after responding; persistent, which works differently from non-persistent, as in this case the TCP connection is not closed, the server waits for a timeout period before closing. The TCP that is used is port 80.

Hypertext Transfer Protocol Secure (HTTPS) is an implementation of HTTP with an additional layer of security that uses Secure Socket Layer (SSL)/Transport Layer Security (TLS). This new layer of security encrypts data and verifies the authenticity of server and client. HTTPS runs over TCP and the port used is 443.

Trivial File Transfer Protocol (TFTP) is a simple file transfer protocol that runs over UDP and does not support any kind of authentication mechanism. The client sends an initial packet to server port 69 and then the server chooses another port for this service, so port 69 can be free for any other requests. Flow control has to be implemented on application level since TFTP runs over UDP and it has no inherent flow control.

File Transfer Protocol (FTP) is a file sharing protocol that runs over TCP using two ports, 20 as data port and 21 as control port (ports may vary depending on the FTP mode). There are two FTP modes: active mode and passive mode. In active mode, the client chooses a random port N for the connection (being N greater or equal than 1023) to FTP server

command port 21. The client starts to listen in port N+1 and then the server connects back to the client desired data port, which is 20. Once data is transferred the server closes the data connection and control connection is only closed when the FTP session ends.
In Passive mode the client opens both connections, being the first on port N (being N greater than 1023) and the second one on port N+1.

Secure File Transfer Protocol (SFTP) runs on Secure Shell (SSH), similar to SSL, offers data encryption and client/server authentication. This protocol runs on port 22.

Web Distributed Authoring and Versioning (WebDAV) is more than a file transfer protocol. It runs over HTTP (there is a new version SSL Web Distributed Authoring and Versioning (WebDAVs) which runs over HTTPS, which uses SSL encryption and is more secure) and is mainly designed for collaboration activities. WebDAV allows multiple users to edit the same file remotely, making this protocol appropriate for enterprises.

Accelerated File Transfer Protocol (AFTP) is a TCP-UDP hybrid that makes file transfer immune to poor network conditions, making file transfer faster without increasing the bandwidth.

Applicability Statement 2 (AS2) is a protocol built for Electronic Data Interchange (EDI) transactions and these exchanges are normally seen in healthcare, manufacturing and retail industries. Data is encrypted using SSL, so it is transported using HTTPS which runs over TCP port 443.

SMB protocol is a file sharing protocol that was implemented by Microsoft Windows. The protocol has various message packets, named dialects. CIFS protocol is one of them. The protocol is a client-server implementation that consists of a set of data packets to establish and terminate connections to shared server resources, access and manipulate files, and send data to print queues, mailslots, and named pipes and provide data about the status of the print queues.

Samba can be used as a file sharing service that uses SMB/CIFS protocol and integrates Linux/Unix servers and desktops into active directory environments. It runs on Unix systems but has no difficulties contacting windows clients. This means that Windows users can access file services without knowing and caring if the service is offered by a Unix host.

2.4 Network security and access control

Network security features are an important aspect when designing a network. Thinking about security when designing reduces or even eliminates future network downtime and expenses. Today's networks are susceptible to security threats such as Viruses, Spyware, Malware, Zero-day, hacker attacks, identity theft, Data theft and Denial of Service attacks (DoS). There is no single solution to deal with all these security threats. Security in a network is built upon several layers and if one fails there will be another to sustain the network. Network security contains components such as Anti-Virus, Firewalls, IDPS and Virtual Private Network (VPN).

Network access control can be done by several mechanisms such as Firewalls, Intrusion Detection System (IDS) and Intrusion Prevention System (IPS).Firewall is a software, hardware or a combination of both that works as a single point of defense to protect the network from

internal or external attacks. It monitors and filters all packets that enter or leave the network by comparing them to the network control policies.

An Anti-virus is basically a software with a list of virus signatures and rules that are used to detect and block viruses. This piece of software is normally installed in the user computer and it can't detect zero-day attacks. Zero-day attacks are new, never seen attacks. Since they are new, anti-virus don't have any signature for these new virus and can't detect them.

IDS and IPS will be discussed in 2.10.

2.5 Attacks to enterprise networks

We live in the era of technology where every new software has a bug that can be exploited by attackers. We will discuss the most common attacks on a network.

Denial of Service (DoS) is a cyber-attack in which the attacker seeks to make a machine or network resource (router, or any network equipment) unavailable to its users by temporarily or indefinitely interrupt services of a host connected to the internet. DoS is typically accomplished by flooding the targeted machine or network resource with a high amount of requests with the end of overloading the system and preventing legitimate requests from being completed.

Structured Query Language (SQL) injection is yet another common attack where the attackers use a data-driven application entry field to input malicious code which will be executed in the victim's server. This attack exploits a security vulnerability in an application software, for example, when attackers input is either incorrectly filtered for string literal escape characters embedded in SQL statements or nor strongly typed and unexpectedly executed. This type of attack is commonly known as a vector attack for websites as a website normally has a database associated and also has input entries which will be executed in the website database.

2.6 Network usage

When using file sharing within the company every user has a unique way of using it, although there can be cases where the usage is the same or simillar. When the company has automated machines that are programmed to do a certain task every day the usage of this machines is probably the same or equal. Since users most of the time have an unique way of using file sharing service, a profile for each user can be created in order to differenciate them and to help on anomaly detection in the network.

We can have different types of file sharing: peer-to-peer where two machines communicate with each other as they share files between them, central server where there is a central machine with files shared by one or more machines and each type of file sharing may have different services associated.

Usually, a company has different types of traffic, such as web, Youtube (video streaming), file sharing traffic such as video editing, creation, edition and deletion of files or folders.

The usage of file sharing service can vary between users. A human is not predictable, so in order to reflect the time between file actions, an exponential variation of a certain meantime

is used. We can have a user that is doing an important task and has the need to save the file with little time between an action, we can have another user that does not require to save in such little time and a user that practically does not do any action. In terms of actions, the action with most probability is editing a file, creation and deleting in not that common. The different user profiles will be explained in chapter 4.

Attacks to the network are considered illicit network usage. There are different type of attacks but we will be focusing on ransomware, theft, and their propagation. The source of the attack can be either internal or external. External attacks normally appear in links or files from social media. People tend to click on everything on the internet and since social media is used by a lot of people, attackers normally propagate their virus using social media or frequently used websites. The attack can also be internal, where the source is someone from inside. Inside attacks might happen if some user connects a pen drive with malicious code or even if someone from the outside gets physical access to a network component.

Ransomware goal is to encrypt everything on a machine making it inaccessible to the legitimate user, while theft simply tries to get hold of all information on a machine and send it somewhere else. Propagation is when the virus is trying to spread to other machines.

Recently, in 2017, there were two major ransomware, WannaCry and NotPetya. WannaCry used an exploit tool known as EternalBlue and NotPetya used EternalBlue and EternalRomance. Both tools were leaked by a group named ShadowBrokers and it is said that these tools were made by National Security Agency (NSA).

Eternal blue exploits three SMBv1 bugs and it is used by WannaCry ransomware to spread the malware. The bugs were named "Wrong Casting Bug", "Wrong Parsing Function Bug" and "Non-paged Pool Allocation Bug".

Wrong Casting Bug takes advantage of a bug on the process of converting File Extended Attributes (FEA) from Operating System 2 (OS2) structure to NT structure by Windows SMB implementation, which leads to a buffer overflow in the non-paged kernel pool.

There are two important commands that must be explained before explaining the Wrong Parsing Function Bug, which are SMB_COM_TRANSACTION2 and SMB_COM_TRANSACTION2. SMB_COM_NT_TRANSACT belongs to a set of sub-commands that provide support for a richer set of server-side file system semantics. These commands allow clients to set and retrieve Extended Attribute key/value pairs, use long file names, perform directory searches and other tasks. SMB_COM_NT_TRANSACT sub-commands offer an extension of file system feature access provided by SMB_COM_TRANSACTION2. In both SMB_COM_TRANSACTION2 and SMB_COM_TRANSACTION2_SECUNDARY, the maximum data that can be sent is represented in parameter "Word". On SMB_COM_NT_TRANSACT the field that represents the maximum amount of data is "Dword". This "Dword" field is also in SMB_COM_TRANSACTION2_SECUNDARY, so there is a difference between the amount of data that can be sent in SMB_COM_TRANSACTION2 and in SMB_COM_NT_TRANSACT. There is no function that validates which function started the transaction, if it was SMB_COM_TRANSACTION2 or SMB_COM_NT_TRANSACT,

which can lead to wrong data parsing by treating Dword as Word which enables Wrong Casting Bug that we talked before. The fact that there isn't a function to validate which function started the transaction is the Wrong Parsing Function Bug. Non-paged Pool Allocation Bug takes advantage of the fact that there is a way to allocate a chunk with a specific size in the kernel non-paged pool that is used in the heap grooming phase creating a hole that can be filled with a data size that causes out of bound write to the next chunk.

In brief, in the eternal blue exploit, the attacking machine tries to discover every machine on the network, sending ARP requests. After getting the response from the machines it tries to verify if the machine is vulnerable to the exploit, and if it is, starts to communicate and do the exploit. After the communication, the victim machine will be infected and the process goes on.

2.6.1 Anomalies

Anomalies are instances of data captured from the network that do not conform to a well-defined notion of normal behavior. Instances can be called objects, points, events, vector, or samples by various authors. Network anomalies refers to finding non-conforming patterns or behaviors in network traffic data. This non-conforming behaviors are often called as anomalies, outliers, discordant observations, exceptions, aberrations, surprises, peculiarities or contaminants in different application domains. [1]

The illicit usage of the network that we talked before in 2.6 are considered anomalies, but it is possible that a licit usage of the network is also considered an anomaly. For example, assuming that max upload in a 2min time space to a central server is 50Mb, there can be a case of a user uploading 100Mb. (Bigger project upload or another normal case). In this case the operation was licit but it wasn't a normal operation to do and so it is considered an anomaly. The machine must be trained well in order to detect if an anomaly is licit or illicit and block only the illicit.

Scenario 1

Since EternalBlue uses an SMB vulnerability to propagate to other machines we can use this exploit as a testing script of a propagation malware anomaly.

Lets say that in a normal use case scenario, we would see 2 to 8 machines connect to the network at the same time, so we could see 2-8 arp requests uploaded by a machine.

An anomaly would be a higher amount of arp request from the infected machine to other IP addresses.

Scenario 2

In this scenario we have the infected machine and a central internal server with sync, which means that the files on the user machine are synchronized using some sort of desktop application (example: ownCloud). To represent a normal human behavior we will use an exponential distribution function with three different means representing one of each normal profiles.

In the illicit usage case, in a Ransomware attack the infected machine will try to delete the server files and replace them with new encrypted files making them inaccessible. In case

of ransomware, we will see a higher amount of uploads and downloads. Also in ransomware, we will detect that its behavior is monotonous. In case of theft we will see a higher amount of uploads to an external IP address.

Scenario 3

In this scenario we use an external server and a user machine that wants to access the files from the server.

The anomalies are similar to scenario 2, but in this case the connection to the file sharing service is from an external IP.

2.7 Data acquisition

In order to analyze all the data that passes through the network, we need some kind of device that can get access to all data that passes within the network. There are some options such as network Terminal Access Point (TAP)'s and port mirroring.

A network tap is a direct connection between a sensor and the physical network media itself (example: fiber-optic). A TAP monitors events on a local network and helps network administrators in analyzing the network. The TAP is normally a hardware device that is normally located between two points in the network, let's say A and B. So, in this case, the network TAP will have at least three ports, one A port, one B port, and one monitoring port. A TAP inserted between point A and B passes all traffic (sends and receives data), in real time, but it also copies all data to the monitoring port. This enables a third party, let's say an admin that needs to monitor the network, to listen to the flow between point A and B. Network TAP's are useful for network intrusion detection systems, VoIP recording, network probes, Remote Network MONitoring (RMON) probes, packet sniffers, and other monitoring or network flow collectors. TAP is a good option because they are non-obtrusive, not detectable (no physical or logical address), deal with full-duplex and non-sharing networks, and pass through or bypass traffic even if the TAP stops working.

A port mirroring is another option to gather network traffic. It is used in a network switch to send a copy of all network packets that pass through one switch port to another switch port for monitoring purposes. This option is used in network intrusion detection systems, network passive probes, real Real User Monitoring (RUM), packet sniffers, and other monitoring or network flow collectors like TAP's.

2.8 Machine Learning

Machine Learning (ML) is the science of programming a machine in such a way that it can learn from data without being explicitly programmed.

"A computer program is said to learn from experience E with respect to some task T and some performance measure P, if its performance on T, as measured by P, improves with experience E." [2]

ML is a subfield of Artificial Intelligence (AI) that has evolved from pattern recognition, used to explore data structures and fit into models that can be understood and utilized by

users. ML can be seen as a program that is constructed using historical data, to solve a given problem, and automatically improving its efficiency with experience. Machine Learning has been used in the past few years in solutions such as detection bank frauds, information-filtering systems (example: email spam), autonomous vehicles. There are a lot of technologies that take advantage of machine learning: recommendation engines, that suggest which movies or tv series to watch next; Optical Character Recognition (OCR) technology converts text images to actual editable text; Self-driving cars, as the name indicates, are self-driving cars that learn from experience; information filtering, such as email spam, that filters all emails that are flagged as spam.

ML is classified into supervised, unsupervised, semi-supervised, and reinforcement learning. A supervised task performs both classification and regression. Classification is the prediction of the nominal class label, where regression is used to predict the numeric value of the class label.

ML can be applied in the detection of these anomalies because we can use ML properties of automatic improvement of its efficiency by experience and the fact that it uses historical data when constructed. So, we can create a machine with both normal and abnormal behavior in an enterprise network and then use it to detect anomalies and improve its efficiency in the meantime.

We can take two approaches into our problem, detecting anomalies by classification or by detecting outliers. In classification, we have a clear view of what anomalies we want to detect and also know their behavior. We train the machine with normal behavior and the anomaly behaviors we want to detect, having two labels, normal and anomaly. So, any anomaly that is known by our ML algorithm will be detected.
In the detection of outliers approach, we train the ML algorithm with the normal behavior, it will then create a boundary and any new sample that is out of the boundaries (outlier) is considered as an anomaly.

In the classification approach, we know what we want to detect and every anomaly with a similar behavior will probably be detected. The downside is that any anomaly that is not known by the ML algorithm won't probably be detected (example: zero-day attacks). But in the other hand, every anomaly that is trained will be detected and it will have low false positives.
In the outlier approach, everything that isn't considered as normal will be considered as an anomaly, in this case, we can probably detect all anomalies (also zero-day). The downside of the outlier approach is that any change of the normal behavior can be considered as an anomaly but in reality it is not, creating high false positives.

We will test both approaches and study which one of them is better.

2.8.1 Types of Machine Learning

There are different types of ML based on: whether or not the system is trained by a human (supervised, unsupervised, semi-supervised, and reinforcement learning); ;whether or not they can have an incremental learning ;whether they work simply by comparing new data points

to training data points or if they detect patterns in the training data and build predictive models.

ML systems can be classified into four categories depending on how much supervision they get during the training. Those four categories are supervised, unsupervised, semi-supervised and reinforcement learning.

Supervised learning is a learning model that is built to make a prediction, giving an unforeseen input. The supervised learning algorithm takes input dataset and its output to create a regression/classification model. This model is then used to predict new data. Supervised learning has known algorithms such as linear regression, logistic regression, neural networks, decision trees, Support Vector Machines (SVM), random forest, naive Bayes, and k-nearest neighbor. Classification tasks try to categorize/tag/label the data into two or more groups or classes, while Regression predicts continuous responses by labeling data into two different classes. Let's say, we have a dataset of people that drink tea and people who doesn't and their time of sleep. In this case, we have drink and sleep and two different variables. With Regression, we can relate both variables and do predictions.

In unsupervised learning, the training data is not labeled. In unsupervised, the goal is to find the regularities in the input data, such as patterns, and find out what normally happens, creating a model. There are different unsupervised algorithms such as clustering, visualization and dimensionality reduction, and association rule learning.

An unsupervised task is dimensionality reduction, aiming to simplify the data without losing too much information. One way of doing this is to merge correlated features into one Principal Component Analysis (PCA). Another unsupervised task is anomaly detection, where the system is trained in normal instances and when a new instance does not belong to the normal instances it is considered an anomaly.

We know that supervised learning deals with labeled data and that unsupervised learning deals with unlabeled data. In the semi-supervised learning, we have the combination of both labeled and unlabeled data. Most of the semi-supervised learning algorithms are a combination of both supervised and unsupervised learning algorithms. The goal of a semi-supervised algorithm is to label all possible data using the supervised algorithm and then try to label the rest of the data using the unsupervised learning algorithm.

Reinforcement learning addresses the problem of an autonomous agent, when interacting with the surrounding environment, to choose the optimal action to achieve its goal. It considers the consequences of its actions and takes great improvement. One example is a machine playing chess with a real player.

2.8.2 Machine Learning algorithms evaluation

In ML, a model is constructed based on the analysis of a training dataset. It is used to predict the class label of objects that do not have any label (unlabeled). An example of a classifier is the decision tree, where each node denotes a test on an attribute value, the branch represents the tests outcomes and the leaves denote each class. There are also binary classifiers, where we have only two classes. Accuracy, confusion matrix, precision and recall, bagging, are some

of the evaluators that we will explain in the next subsections.

Accuracy

The accuracy of the algorithm is defined as the correctly classified instances of the total instances. It's the number of correct predictions to the total number of predictions. For a correct calculation of an algorithm accuracy, the test data must be balanced.

Confusion Matrix

A Confusion Matrix (CM) is a table that saves the number of instances in a dataset and the corresponding category. In a binary training set, we have two possible values, positive class, and negative class. As we can see in 2.5, the number of positive and negative classes that are correctly predicted is called True Positives (TP) and True Negatives (TN), while all the instances that are wrongly predicted are called False Positives (FP) and False Negatives (FN).

		Predicted class	
		Positive	**Negative**
Actual	**True**	TP	FN
Class	**False**	FP	TN

Figure 2.5: 2x2 Confusion Matrix [3]

Precision and Recall

Precision is the fraction of relevant instances from all the retrieved instances.

$$\text{Precision} = \frac{\text{True Positive}}{\text{True Positive} + \text{False Positive}}$$

Figure 2.6: [3]

While in recall, which is the opposite of precision, tries to address the question of how many actual positives were identified correctly.

$$\text{Recall} = \frac{\text{True Positive}}{\text{True Positive} + \text{False Negative}}$$

Figure 2.7: [3]

Train/test split

Train/test split when one splits the dataset into train and test. The train split is used to fit the model and the test is used as the test set. The usual train/test percentage is 80/20 (train/test) or 70/30 (train/test). An example of a train/test split is shown in Figure 2.8 .

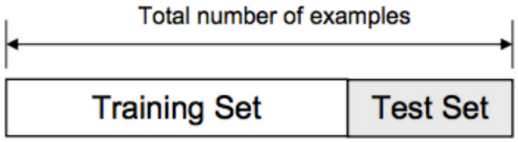

Figure 2.8: Train/Test split [5]

k-Fold Cross-validation

k-Fold cross-validation is a technique that validates the model without reducing the training dataset. In this technique the data is divided into k-folds. One of the folds is used as the validation/test set and the remaining k-1 folds are used as training set. The model evaluation is obtained by calculating the average over the K trials. A representation of this technique can be shown in figure 2.9 .

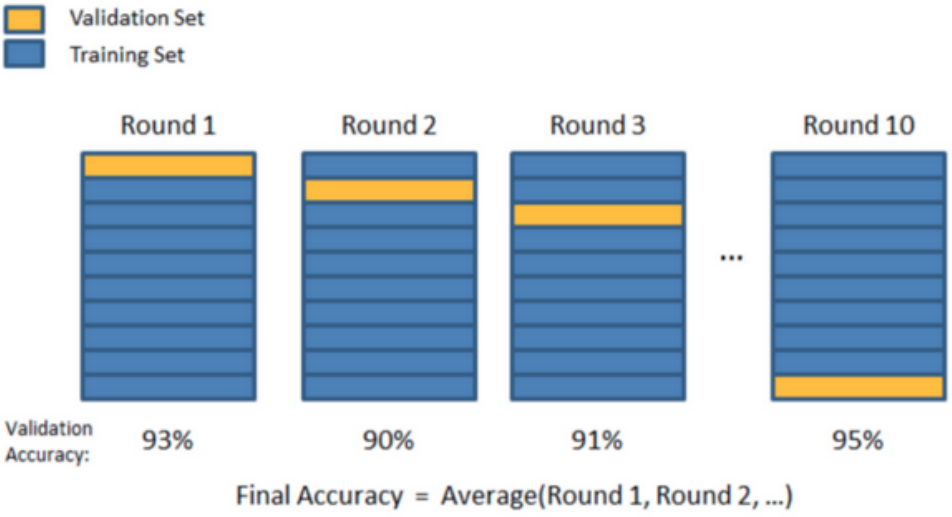

Figure 2.9: k-fold cross-validation [6]

Stratified k-Fold Cross-validation

This technique is similar to the normal k-fold cross-validation but in this case, a fold has equal representation of each class label. This technique might outperform normal k-fold in datasets where class labels are unbalanced.

Bootstrapping and Bagging

Bootstrapping is a technique where the analyst repeatedly draws different samples from the same dataset with the goal of having a better representation of the class labels that are present on the dataset.

[5] `https://towardsdatascience.com/train-test-split-and-cross-validation-in-python-80b61beca4b6`, accessed 28/10/2018

[6] `https://towardsdatascience.com/train-test-split-and-cross-validation-in-python-80b61beca4b6`, accessed in 28/10/2018

Bagging helps to reduce the variance of the models. It is a method for generating multiple versions of a predictor and using these to get an aggregated predictor.

2.8.3 Machine Learning training models

In ML, hyperparameter optimization or tuning is the problem of choosing the optimal set of parameters for a learning algorithm. Hyperparameters are the different constraints, weights or learning rates that a ML model can require to generalize different data patterns. Grid search is traditional way of hyperparameter optimization that simply makes an exhaustive searching trough a manually specified subset of the hyperparameter space of the learning algorithm. Random search is another way of performing hyperparameter optimization, it replaces the exhaustive enumeration of all combinations from grid search by selecting them randomly. Random search outperforms grid search.

Linear regression is one of the simplest models,since it only has a linear function associated. This model makes a prediction by computing a weighted sum of the input features, plus a constant called bias term. A linear function is shown below, where y is the predicted value, O is the model parameter that includes the bias term and the feature weight, x is the feature value and n is the number of features.

$$y = O_0 + O_1 x_1 + O_2 x_2 + O_n x_n$$

Gradient descent is a generic optimization algorithm that is able to find the optimal solution for a wide range of problems. It iteratively tweaks its parameters in order to minimize a cost function. Initially, we give random values to the model and then it improves gradually by attempting to decrease the cost function until the algorithm converges to a minimum. The size of the learning steps that the algorithm does in order to get to the optimal solution (minimum) is determined by the learning rate hyperparameter. If the learning rate is small, the algorithm will take long to converge. On the other hand, if it is too high, the algorithm will diverge with larger values with the possibility of failing to find a good solution.

Polynomial regression is a technique that enables you to fit nonlinear data using a linear model. This is possible by adding 'powers' of each feature as new features and then train a linear model with this extended features.

2.8.4 Regression Algorithms

In ML, algorithms can be divided into parametric and nonparametric learning models. Parametric learning models is when we have algorithms which have strong assumptions in the learning process and simplify the function to a known form. Regression algorithms such as linear regression and logistic regression are examples of parametric ML algorithms. These algorithms work by modeling of the relashionships between the variables that are refined iteratively using a measure of error in the predictions made by the model.

Linear regression is an algorithm that tries to find the optimal fitting straight line through the given data points. This algorithm attempts to model a relationship between two variables by fitting a linear equation to the observed data, where one of the variables is considered to

be an explanatory variable and the other a dependent variable. While linear regression tries to relate two variables with a straight line equation, nonlinear regression generates a line as if every value of the second variable is random. Functions such as logarithmic, exponential and trigonometric are also part of nonlinear regression algorithms.

Logistic regression estimates the probability of an event of being a success or failure. Meaning that the variable must be binary (0 and 1) in order to use logistic algorithm.

Polynomial regression is a form of regression analysis where the relationship between the independent variable x and the dependent variable y is modelled as an nth degree polynomial in x.

2.8.5 Classification Algorithms

As discussed in 2.8.1, classification uses supervised mode in order to build a model. Unlike parametric algorithms, nonparametric do not have strong assumptions or hypotheses about the mapping function. Nonparametric are normally more flexible but require a large amount of data and training time. Algorithms such as SVM, Neural networks, and decision trees are examples of nonparametric algorithms.

Decision tree algorithms predict the target variable based on an input variable by learning decision rules. The prediction of the variable takes several stages. It starts on the first root node, then it follows all the nodes until it gets to the edge, reaching the last node. In each node, it performs an operation. When it reaches the last node, the tree predicts the variable class.

SVM can perform linear classification and also nonlinear by using a method known as a kernel function, which maps the input vector into a high-dimensional feature space. SVM training algorithm builds a model by labeling every sample as one of the two labels/classes. The training algorithm is then capable of deciding which class a new sample belongs to by projecting the sample to the vector space and checking in which side of the line it belongs to.

In Linear SVM, there is a hyperplane that separates the data into two different classes. Let's say we have a set of samples (Xn, Yn), Yn can have the values +1 or -1 which represents the class that the samples are in. The SVM then learns the maximal marginal hyperplane that separates both classes:

$$W.X + W0 = 0$$

W is the weight assigned to a data point, X denotes the input vector. If the result of the equation is greater than zero, the data point belongs to class 1, otherwise it belongs to class -1.

On the contrary, in nonlinear SVM, there is no strong assumption about the linearity of the data. The assumption of the linearity of the dataset is a bit unrealistic as there are cases that data cannot be linearly separated. Here is where nonlinear comes in, where we can apply kernel functions in order to extend the linearity of an SVM into nonlinear SVM. When the dataset cannot be linearly separated, it is mapped into a higher dimensional space and then uses a linear classifier in the new projected higher dimensional space. The projection of the dataset into a higher dimensional space is called "the kernel trick". This kernel trick

makes data linearly separated by exploiting mathematics that projects the data into a higher dimension. An example of a projected dataset into a higher dimension is shown in 2.10.

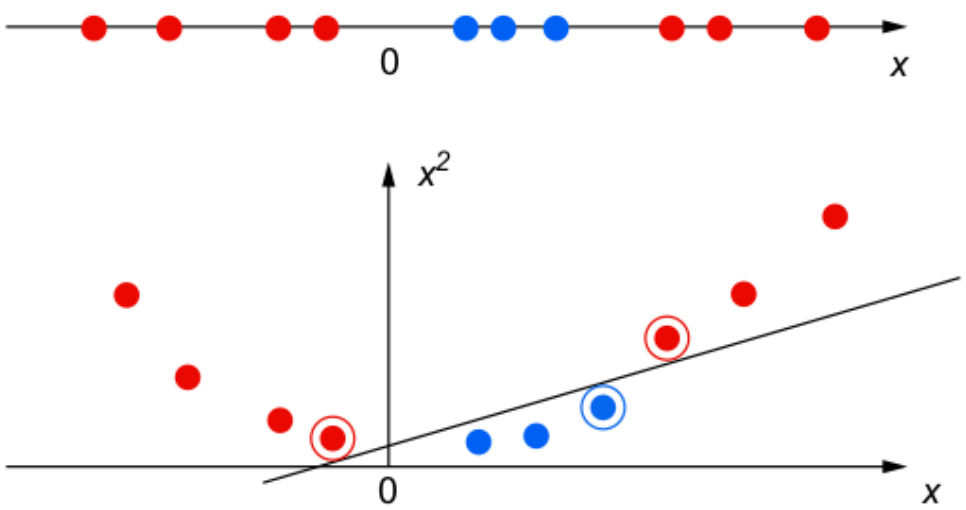

Figure 2.10: Example of a projected dataset [3]

2.8.6 Clustering Algorithms

Clustering algorithms are part of the unsupervised learning algorithms. The input dataset consists of unlabeled data which has no class or target values. It groups data that are similar into the same cluster and data that are unrelated into different clusters. There are many applications of clustering in the real world, being gene sequence study, market research, medicine, psycology, sociology, marketing, biology, some of them. We have many clusters algorithms like k-means, k-medoids, hierarchical clustering, fuzzy c-means.

K-means clustering is where the dataset is divided into k number of clusters. Each cluster has a center named centroid. The algorithm takes the following steps:

Step 1: Choose k random data points that will be the first centroids for each cluster.
Step 2: Assign each dataset data point to the closest centroid.
Step 3: Use the new added points to recalculate each cluster centroid.
Step 4: Check if the union is met, if not, go back to step 2.

Hierarchical clustering groups data into a hierarchy of a "tree" of clusters. Depending on the clustering decomposition, we have agglomerative or divisive clustering. In Agglomerative clustering, partitions are viewed as a tree structure called dendrogram. Divisive clustering is based in the concept that the feature vectors form a single set and then hierarchically divide this group into different sets. In both agglomerative and divisive the number of clusters has to be inputted as a termination condition. The steps for the agglomerative approach are as follow:

Step 1: Each data point is its own singleton cluster.
Step 2: After each iteration of Euclidean distance calculation, merge two clusters with minimum distance.

Step 3: The algorithm stops when there is a single cluster of all examples, if not go back to step 2.

The steps for the divisive approach are:

Step 1: Start with a single cluster with all data points.

Step 2: After each iteration, remove all the outsiders (all points outside the range of the cluster) from the least cohesive cluster.

Step 3: Stop when each example has its own singleton cluster, else go to step 2.

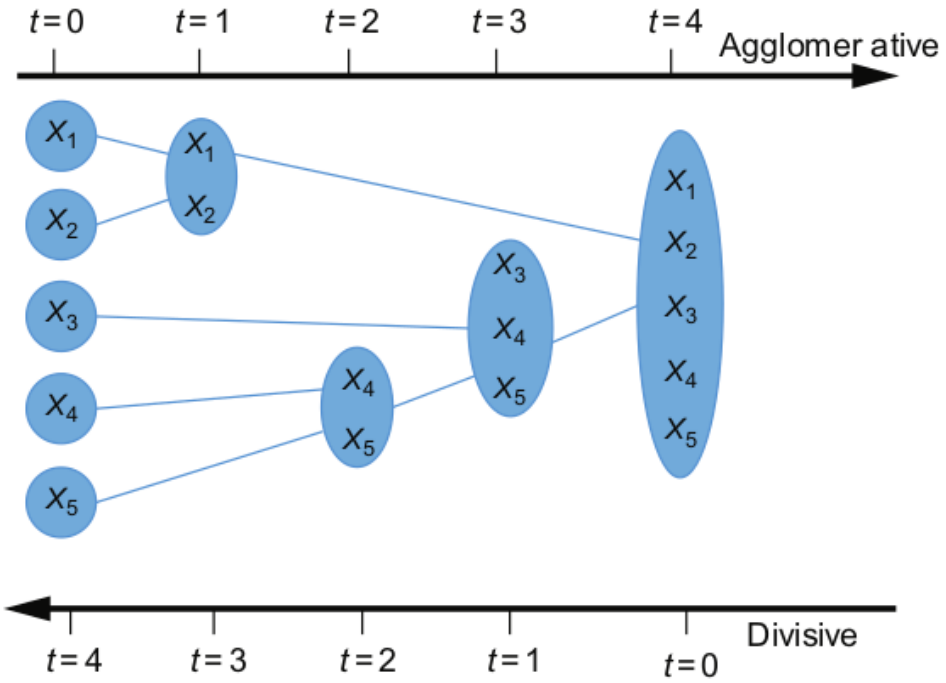

Figure 2.11: [3]

2.8.7 Ensemble learning

Let's say we have a complex question that is answered by thousands of people, their aggregated answer might be better than an expert answer. An aggregation of predictions is called an ensemble. In ML, one can take predictions from multiple algorithms and make a decision taking in mind those predictions. There are severall ensemble methods such as bagging, boosting, stacking.

Voting Classifiers

Supposing that we have multiple classifiers like logistic regression, svm , random forest, and some other where all of them cannot achieve an accuracy higher than 80%. A good way of getting a better classifier is to aggregate all of the classifiers predictions and predict the class that has more votes. This method is called the hard voting classifier.

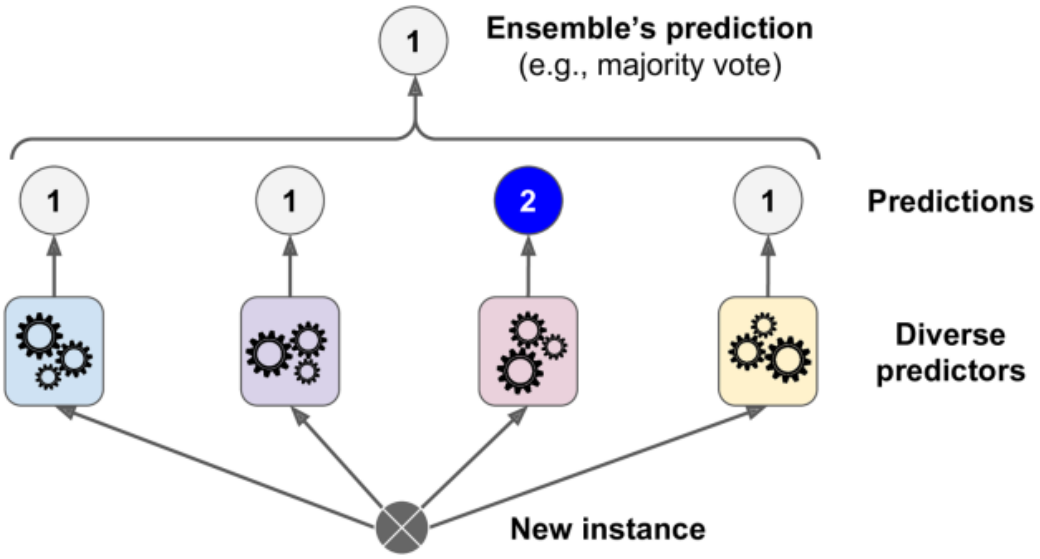

Figure 2.12: Hard voting classifier [4]

2.9 Network business environment

When designing a network, there are some design considerations that should be taken into account:

- Standards - network components should be based on open standards as they give higher flexibility when interconnecting components from different vendors.
- Scalability - introducing new networks components shouldn't require the re-design of the network topology.
- Availability and Reliability - time between failures and repair should be considered while designing the network.
- Modularity - the network should be divided into sub-systems for better management and to isolate them in case of failure.
- Security - security is one of the most important topics as we have to ensure that the network have no holes for attackers to exploit.
- Network Management - provides a way of monitoring the network and take actions.
- Performance - should be taken into consideration in order to have good network performance.
- Economics - there should be a balance between cost and performance

An example of an enterprise network is shown in 2.13.

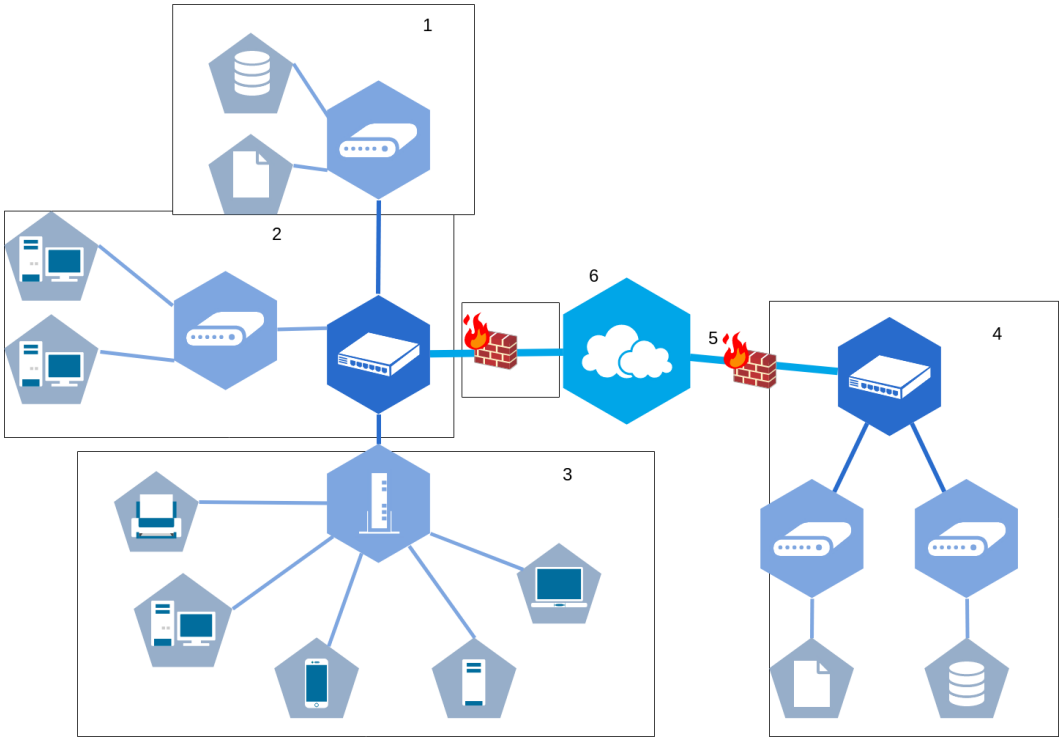

Figure 2.13: Enterprise network example

Number 6 represented on 2.13 is a representation of the internet. Number 5 is the firewall. Firewall is a software, hardware or a combination of both that works as a single point of defense to protect the network from internal or external attacks. It monitors and filters all packets that enter or leave the network by comparing them to the network control policies.

Number 4 represents enterprise external servers with various services such as file-sharing, database and file storage.

Number 3 is a tipical "public zone" devices are connected to the enterprise network.

Number 2 is one example of two working employees computers connected to a network switch. This case can be replicated multiple times. Replication is normally done to represent VLAN.

Number 1 is the internal enterprise server which can contain database, file-sharing service, file storage and more services associated.

2.10 Intrusion Detection and Prevention Systems

Computer network security has three main fundamentals: prevention, detection, and response. Lately, prevention and detection were the most researched fundamentals because if we detect and prevent all security threats there is no need for the response.

An intrusion can be defined as: "Any set of actions that attempt to compromise the integrity, confidentiality, or availability of a resource is termed an intrusion" [5]. Intrusions are basically abnormal behaviors that normally start from an outside attacker who wants to gain unauthorized access in order to obtain sensitive information.

Intrusion Detection (ID), in the networking context, refers to the detection of malicious activities such as illegal access, penetrations or other forms of computer abuse.

Intrusion Prevention (IP) prevents an illegal access to a system resource.

2.10.1 Network-based Intrusion Detection System

For the detection of these malicious activities, intrusion detection systems process and analyze a large amount of data. An IDS in a network environment is not a new technology, as it has been used by companies and organizations in order to protect valuable resources.

The misuse-based technique is based on a set of rules or signatures of known attacks. A signature, in computer networks, is a characteristic left by a cyber attack. Misuse-based techniques can detect attacks by comparing the attack signature with those stored in a signature database. However, having a database of all different possible attacks is a challenging task.

Two major problems can arise while using this technique: attacks that are not on the signature database won't be detected, and the system won't be able to detect new attacks which are not on the signature database.

The anomaly-based technique takes as the assumption that all intrusive activities are necessarily anomalous. A normal activity profile is built to check if the system state deviates from the established normal profile. All deviations from the normal profile are considered as anomalies. This technique has two problems: anomalous activities that are not intrusive are classified as intrusive resulting in false positives and intrusive activities that are not anomalous are not flagged as intrusive but they actually are, resulting in false negatives. To prevent false positives, threshold levels are selected in order to reduce their occurrence.

The hybrid technique takes advantage of both misuse and anomaly techniques as it tries to detect known as well as unknown attacks.

Nowadays, researchers tend to research more on anomaly-based techniques since they can detect known as well as unknown attacks, where misuse techniques only detect known attacks.

Depending on where an IDS is deployed, it can be classified as Host-based Intrusion Detection System (HIDS) or Network-based Intrusion Detection System (NIDS).

A NIDS provides well-defined mechanisms, which collect and analyze data from hosts and network components in order to identify possible intrusions in the network.

2.10.2 NIDS Architecture

A NIDS consists of several parts that must work together in order to produce an alert.

Network Tap is a part of the network where all traffic passes through it (example: router). This network component is important to intrusion detection because we are able to get every network data which is a key aspect for a good detection.

Network Sensor is a software that runs on dedicated machines or network devices and separates normal traffic from suspicious traffic. Anomaly-based techniques can be used to detect anomalies, while Signature-based techniques (or misuse) can be used to detect attacks based on signatures.

Analyzer is a software that receives data from the sensors and determines if the traffic is either normal or anomalous.

Alert Notifier is the network component responsible for notifying the system administrator when the threat level is high enough to be reported.

Manager is a software that is used to manage the whole system. This software is accessed remotely in order to manage the system from any location. It contains a set of tools for setting policies and processing alarms.

Response is a subsystem that provides capabilities to take countermeasures when the system detects an intrusion. These countermeasures can be done by the system administrator or even the system itself and can include router or firewall reconfiguration or in the worst cases shutting down a connection.

Database is a knowledge repository to store all information observed by the intrusion system. This can include either anomaly or signature statistics.

2.10.3 Host-based Intrusion Detection System

A HIDS, as the name implies, is deployed on a host where it monitors and analysis the interior of a system rather than its external interfaces. HIDS is used to detect host intrusions by monitoring system-specific, system, event, and security logs. When a change is detected, the HIDS compares the new log entry with its signature database and determines if it is a normal or a suspicious activity.

2.11 Intrusion Prevention System

An IPS comes as an additional security line, where it detects and prevents intrusions. A Network-based Intrusion Prevention System (NIPS) is able to deny hostile traffic while allowing legitimate traffic into the network. NIPS are usually placed behind a firewall, router or switch, so every traffic has to pass through it. Like IDS, IPS can be classified as host-based and network-based depending on its location.

Host-based Intrusion Prevention System (HIPS) is deployed on a host to prevent the entry of attacks directed to the host. Like HIDS, HIPS monitors and analyzes the operation system for any abnormal behavior.

NIPS is deployed on the network to monitor all in-out network traffic in order to detect and prevent attacks. For the correct function of NIPS, all traffic must pass through it.

Distributed Intrusion Detection and Prevention System (DIDPS) is yet another method of detecting anomalies. By distributing host monitoring agents into intelligent points of the network can reduce computation time and give better response time. The agents distributed on the network are used to collect data, analyze it and send anomaly data to the central and main system for further analysis. [6]

2.12 Types of ANIDS

Anomaly-based network Intrusion Detection System (ANIDS) is an IDS located on a network that uses anomaly-based techniques to detect anomalies in the network by constantly monitoring network traffic data and classifying it as normal or anomalous.

Based on the availability of labeled data, ANIDS can be divided into supervised, semi-supervised, unsupervised and hybrid.

2.12.1 Supervised ANIDS

A supervised approach assume the availability of a training data set which has labeled instances for both normal and anomaly classes. Typically, a predictive model for both normal and anomaly classes is built in order to have a clear view of what is a normal or an abnormal behavior. A new instance is compared to both classes to determine which class it belongs to. One of the problems that may appear when using a supervised approach is the fact that the anomaly class might have limited or insufficient data. Intrusions are becoming more and more sophisticated and the anomaly class needs to be updated in almost real-time in order to capture zero-day intrusions. [1]

2.12.2 Semi-Supervised ANIDS

In the semi-supervised approach the training data set belongs only to normal class. Since there is no training data for the anomaly classes, new instances are compared to normal classes only and an instance that is not from a normal class is clear to be an anomaly. This approach is easier to compare to new instances since there is only one class for comparison. Semi-supervised approach is likely to be more effective when catching zero-day intrusions because there is no data for abnormal classes and everything that it isn't a normal behavior will be considered as an anomaly.

2.12.3 Unsupervised ANIDS

An Unsupervised approach does not require any training data set. This technique assumes that normal instances are far more frequent than the anomaly instances. In general, the idea of normal being more frequent than abnormal instances might be true but when this assumption is not true, this approach may suffer from high false alarm rates.

2.12.4 Hybrid ANIDS

An Hybrid approach combines both supervised and unsupervised . By combining both techniques, this approach can detect known instances by comparing them to a training data set (supervised) and every instance that isn't on the trained data set is going to be handled by unsupervised technique in order to identify a new normal or anomaly instance. Example of an Hybrid ANIDS is illustrated on 2.14.

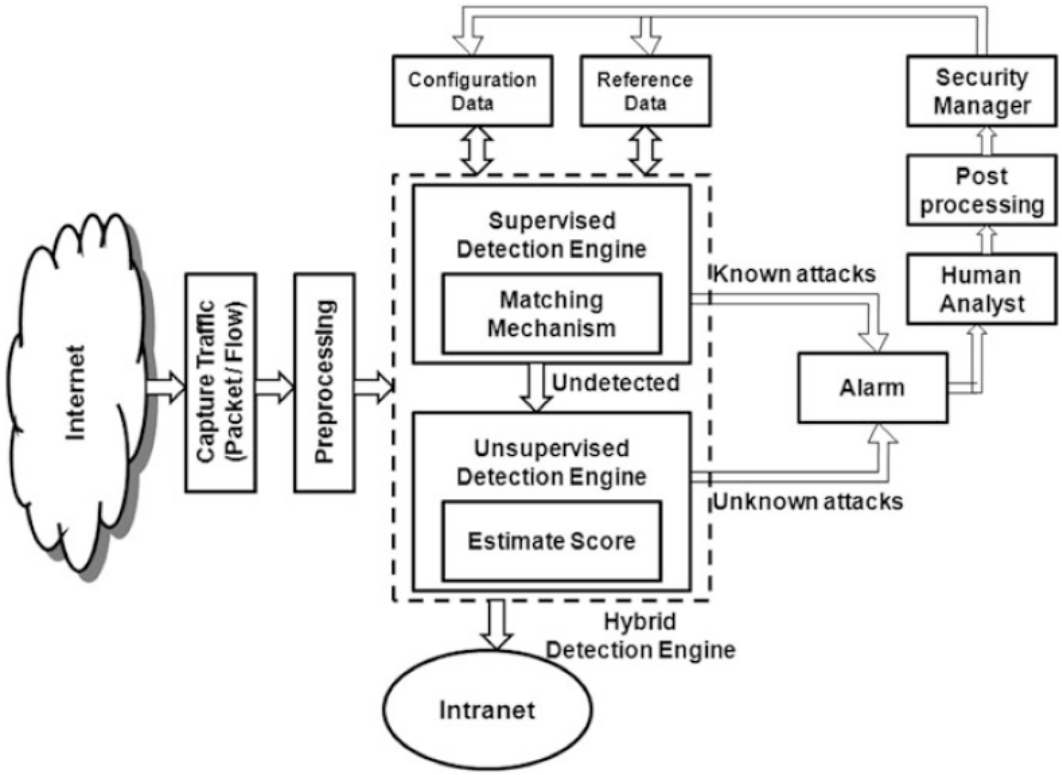

Figure 2.14: Image of an hybrid ANIDS [1]

2.13 Input Data

A fundamental aspect of any anomaly-based network detection technique is the kind of input data that is used for analysis. Input is generally a collection of data instances, also referred to as objects, record, points, vectors, patterns, events, cases, samples, observations, entities. [7] Each data instance can be described using a set of attributes, also referred to as variables, characteristics, features, fields or dimensions. Attributes can belong to different types such as binary, categorical or continuous. Each data instance may consist of only one attribute (univariate) or multiple attributes (multivariate). [8] Input data must go through feature selection and reduction to improve the quality of the data that is going into the system. I will talk about feature selection and reduction in 2.15.

2.14 Data labeling

Data labeling is used in NIDS to help the system distinguish between data instances. These data instances can be labeled either as normal or anomalous. We should keep in mind that obtaining accurate labeled data for both normal and anomalous types is often very difficult and expensive. Labeling is often done manually by human experts and a huge effort is needed in order to obtain a training dataset. [9] Furthermore, an anomalous behavior is often dynamic, which means that new types of anomalies may appear for which no labeled training data exists.

2.15 Feature selection and reduction

Computing raw input data requires large amount of storage and computation time. What is often used in a IDS is feature selection and reduction. Feature selection is an important aspect of the anomaly detection domain. Its role is to eliminate unimportant or irrelevant features from data. Feature selection helps to reduce computational complexity, removes information redundancy, increases the accuracy of the detection algorithm, facilitates data understanding (reducing features,combining features), and improves generalization. The feature selection process includes three major steps: (a) *subset generation*, (b) *subset evaluation*, and (c) *validation*. Three different approaches used for subset generation are: *complete*, *heuristic*, and *random*. Evaluation functions are categorized into five [10] distinct categories: *score-based*, *entropy* or *mutual information-based*, *correlation-based*, *consistency-based*, and *detection accuracy-based*. Simulation and real-world implementation are two ways to validate the evaluated subset.[1]
Feature selection algorithms have been classified into three types: *wrapper*, *filter*, and *hybrid* . Wrapper methods try to optimize some predefined criteria with respect to the feature set as part of the selection process. On the other hand, filter methods rely on the general characteristics of the training data to select features that are independent from each other and are highly dependent on the output. A hybrid feature selection method attempts to exploit the salient features of both wrapper and filter methods. [11]

Feature reduction is also important on the detection of network anomalies. One of the methods used in feature reduction is PCA. PCA uses an orthogonal transformation to convert a set of observations of possibly correlated variables into a set of values of linearly uncorrelated variables called principal components. Each principal component is a linear combination of the original variables. The transformation is defined in such a way that the first principal component is a single axis in space. When you project each observation on that axis, the resulting values form a new variable. The second principal component is another axis in space, perpendicular to the first.[12]

CHAPTER 3

Related Work

3.1 Network Anomaly Detection and Machine Learning

As already said in 2.8, ML is the science of programming a machine in order for it to learn from data without being explicitly programmed. This concept can be useful when studying detection of anomalies in network file sharing as ML provides different classes of learners for anomaly detection and classification. For example, we can make the machine learn what is normal and what is considered an anomaly and make it detect anomalies in real time by comparing the training samples with real-time samples.

We will now study how authors applied different ML methods in network anomaly detection. In this chapter, we will focus on the different approaches that were used in order to detect network traffic anomalies. By studying authors approaches we can divide traffic anomaly detection techniques into Classification, Statistical, Ensemble, Clustering and outlier, Knowledge, Artificial Neural Networks (ANN) and others. "Others" will focus on approaches that apply more than one technique or model.

3.2 Classification based

Classification technique tries to find a boundary between two or more features in order to have a classification of the different features. In computer networks, these techniques try to establish an explicit or implicit model that enables the categorization of network traffic patterns into several classes. Most common classes are normal and anomly. [1] Several classification-based techniques such as k-nearest neighbor, SVM, and decision trees have been applied to network traffic anomaly detection.

Reference [13] introduces a technique that uses decision trees with protocol analysis. This technique builds an adaptive decision tree for each application layer protocol. Data is classified into two classes: benign and anomalous or malignant. Reference [14] uses one-class classifiers that can detect new anomalies, which are data points that do not belong to the learned class. Reference [15] introduces a one-class SVM classifier in which the training data belongs only

to one class. During training, the learning goal is to determine a method which is positive when applied to points within training points boundary and negative outside. This kind of SVM is also used in outliers technique.

In 2016 reference [16] introduced an optimization method for parameters of SVM in NIDS. SVM performance is directly related to its parameters, C, and kernel function, g. If the parameters choice is not proper, the accuracy of the detection accuracy is low. Generic Algorithm (GA) and Particle Swarm Optimization (PSO) are used to obtain optimal parameters. Since using the basic PSO is not enough, because the local search ability is weak, the authors proposed an Improved Particle Swarm Optimization (ICPSO) which uses chaos operator ergodicity, randomness, sensitivity to initial conditions and other characteristics to optimize SVM parameters. This algorithm introduces chaos into the inertia weight factor parameter and improves PSO convergence speed and precision. Module architecture can be seen in 3.1.

Firstly, the selected network data (dataset) is processed by the discrete processing which divides the data into training and test sets. The training data is used as the SVM learning sample. Their improved ICPSO algorithm iterates and updates particle position with the goal of obtaining optimal SVM parameters. The SVM then uses the optimal parameters and outputs the detection results. They concluded that the experiment was effective but there was still some distance to meet the actual network complexity.

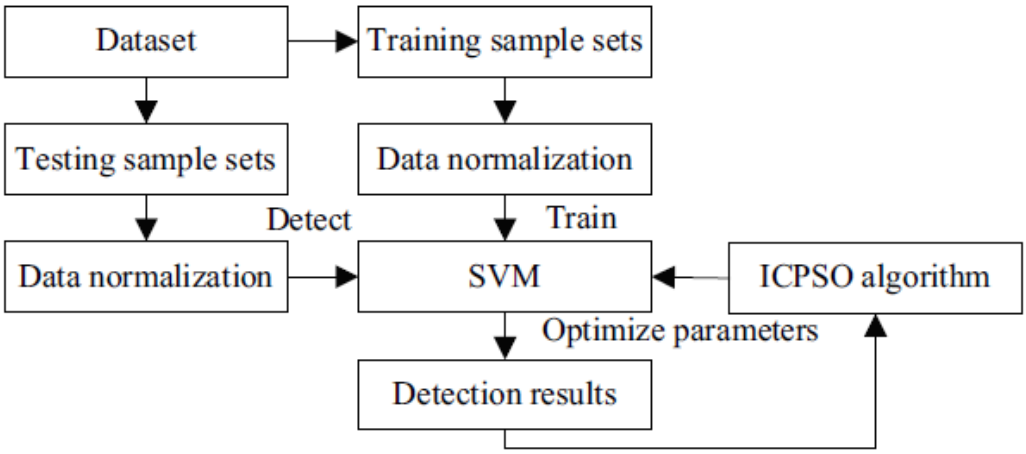

Figure 3.1: ICPSO-SVM architecture [16]

Port scan is a process that sends requests to a range of server port addresses on a host, with the goal of finding an active port. This process is commonly used by attackers to gain access to a computer. Reference [17] tries to detect network port scanning by using anomaly detection. First, they use static tests to analyze traffic rates. Then, they use two dynamic chi-square tests in order to detect anomalous packets. Additionally, they model network traffic as a marked point process and introduce a general portscan model. The authors present true positive simulation tests greater than 90% and false-positives lower than 15% for both static and dynamic tests. Reference [18] uses ML techniques in computer network intrusion

detection that aids analysts to classify network events. Tree classifier algorithms are used to determine which set of attributes most accurately categorizes the data. The following algorithms were tested: ADTree, J48, C4.5, LADTree, BDTRee, RandomForest, RandomTree, REPTree. AAlternating Decision Tree (ADTree) consists of decision nodes that specify a predicate condition and prediction nodes that contain a single number. C4.5 [19] is a statistical classifier that constructs decision trees using information entropy. A pruned or unpruned C4.5 decision tree is generated using a J48graft decision tree. NBTree was used to scale large databases and outperform decision trees. RandomTree is formed by stochastic processes. Some examples are Random Recursive Tree, Random Forest. RandomForest contains multiple decision trees and the output of each tree is the mode of the classes.

The labels used were '1' to anomaly and '0' to normal class. The tree based classifier values were the default ones. The model has the 8 classifiers that we talked before. They concluded that RandomTree, RandomForest, and REPTree detect anomalies powerfully with fewer false rate alarms when compared to other algorithms.

3.3 Statistical based

An anomaly, in the context of a statistical anomaly detection, can be described as: "An anomaly is an observation which is suspected of being partially or wholly irrelevant because it is not generated by the stochastic model assumed". [20]
With this assumption, we can assume that normal data instances occur in high probability regions of a stochastic model and anomalies occur in the low probability regions of the stochastic model. Statistical techniques fit a statistical model (usually done for normal data) to the given data and then apply statistical inference tests to determine if a new instance belongs to the model. Instances that present low probability on the given test are considered to be anomalies.[1]
Assuming that the network traffic follows a certain distribution, statistical techniques are normally designed based on network traffic distribution. The most straightforward way of building a statistical model is to compute the parameters of a probability density function for each known class of given data and then test new instances to determine which class it belongs to [21]. Normally, when generating a probability density function, two different profiles are built, one for normal and another for abnormal behavior.
Essentially, to estimate probability density function, there are two main techniques, parametric and nonparametric [22]. The main difference between both parametric and nonparametric is that parametric techniques assume the knowledge of the underlying distribution and with that knowledge estimates parameters from the given data [23]. On the other hand, nonparametric techniques, normally do not assume the knowledge of the underlying distribution [22]. On nonparametric techniques, the density function is derived based on the network traffic distribution and parameter estimation. This makes nonparametric techniques more flexible than parametric techniques because nonparametric do not depend on underlying knowledge. [1]

Reference [24] introduces a hierarchical multi-tier multi-window statistical anomaly detection system that uses statistical models and multivariate classifiers. This method can detect network anomalies with traffic anomaly intensity as low as 3-5 % of the typical background traffic intensity.

Reference [25] present a nonparametric adaptive Cummulative Sum (CUSUM) method that consists of two key features: its detection threshold can adapt itself to various traffic conditions and it can timely detect the end of an abnormal behavior within a required delay. Reference [26] proposed a multivariate probabilistic calibration model for anomaly detection and localization. This technique applies latent variable probability theory with multivariate t-distribution to establish the normal traffic model. The proposed solution consists of anomaly detection and localization. To detect if a sample traffic is an anomaly, the Mahalanobis distance is compared to a treshold and if it exceeds this treshold it is considered an anomaly. To locate the anomalies the author applies contribution analysis to the samples and locates the anomaly by intersecting the rows and columns of traffic matrix X, where row X corresponds to the time of the anomaly occurence, and column X to the position where the anomaly occurs.

Reference [27] introduces a payload-based anomaly detector named Payload-based anomaly detector (PAYL). During the training phase, PAYL computes a profile byte frequency distribution and their standard deviation of the application payload flowing to a single host and port. For the payload model, PAYL uses n-Gram. N-gram is the sequence of n adjacent bytes in a payload. PAYL models the payload using n-gram analysis and n=1 when it is the byte value distribution. It then counts the occurrences of each n-gram by passing a n width sliding window over the whole payload. The feature vector of a payload is the relative frequency count of each n-gram, which is calculated by dividing the number of occurrences of each n-gram by the total number of n-grams. During the training phase, PAYL observes many payloads and then compute their mean and variance of the byte value distribution, producing a model. During the detection phase, each new payload is scanned and its byte distribution value is computed. The new payload distribution is then compared to the trained models and if the new distribution payload is significantly different from the norm, the detector flags the packet as anomalous and generates an alert.

Reference [28] introduces Network at Guard (N@G) which is a hybrid IDS having capabilities for both NIDS and HIDS that aims to reduce false alarms. The STATistical anomaly detection (STAT) module is implemented in three phases: Initial Learning, Detection, and Continuous learning. The Chi-Square technique is used to compute values for identified parameters, setting thresholds and detecting the deviations from these set of thresholds. It has four detection methods: traffic capturing, network access policy, signature-based detection and protocol anomaly detection. The challenges and issues of this approach were fastness of matching packets against their signature database and processing packets at a fast rate; signature representation flexibility; effective boundaries of signature and the impact in false alarms; Signature farming and chaining which represents the identification of a chain of signatures for a single attack.

Reference [29] introduced Flow-based Statistical Aggregation Schemes (FSAS) for Network

Anomaly Detection. FSAS uses the concept of IP flow to develop its flow-based aggregation technique. FSAS has two main modules: feature generator and flow-based detector. Feature generator has four sub-modules: flow management, probe, feature extraction and an event timer. Flow management module decides if a new packet is part of an existing flow or if there is a need to create a new flow. Probe collects network traffic and abstracts network traffic into a set of statistical parameters that reflect the network status. Feature extraction extracts the features that describe flows behavior. Event timer periodically calls the feature extraction module to convert statistical information of flows into the format required by the statistical model. The flow-based detector has two sub-modules: feature scoring metrics and neural network classifier. Feature scoring metric calculates the probability scores of features by comparing them with the reference model. The Neural network classifier prioritizes classification of score vectors of malicious behavior. Reference [30] proposes a Context-aware Packet Filter Scheme that uses a generated blacklist using statistical-based technique. This architecture has two main modules: context-aware packet filter and monitor engine. Context-aware packet filter is composed by blacklist packet filter which compares packet payloads with signatures according to the IP addresses. Blacklist packet filter contains two modules: Blacklist and Look-up Table. The Blacklist contains all blacklisted IP addresses, this list is updated periodically by monitor engine. Look-up Table contains Matched NIDS signatures and all NIDS signatures. The Monitor Engine goal is to collect statistical data from NIDS and to calculate the confidences of the IP addresses. It also updates the blacklist in a fixed time slot.

3.4 Ensemble based

The ensemble-based technique weights and combines several classifiers to obtain a single classifier that outperforms every combined classifier. This technique allows the combination of outputs produced by basic classifiers that are trained on different subsets of the dataset. Those outputs, weighted combination of multiple classification models, are then sent to the ensemble classifier, which is a predictive model, to do the prediction. Ensemble-based methods can be divided into bagging, boosting and stack generalization. Bagging combines classifiers outputs in order to increase classification accuracy. Boosting takes misclassified instances from previous models and incrementally builds an ensemble classifier. Stack generalization uses output probabilities from each class label classifiers achieving high generalization accuracy.

Reference [31] proposed an intrusion detection ensemble-based technique using SVM classifiers as the component learner. The model includes a feature selection model (DP1) and sample reduction model (DP2). The authors ensemble approach included three steps, the first being data processing where connection records are translated into custom readable data. The second is the construction of different connectional models to achieve better generalization performance and the last step was to form a final decision with the outputs of every other classifier. Their ensemble approach uses Weight Voting Rule (WVR), which is used to make the final decision of the set of decisions from every classifier. They also use a CM to store the accuracy from each classifier for each data class.

Reference [32] evaluated different classifiers and evaluated their performance. The studied ensemble algorithms are Bagged tree, AdaBoost, LogitBoost, GentleBoost and RUSBoost. The process is divided into learning and prediction. In the learning process the training set is divided into different samples, which are then divided into different classifiers. In the prediction process, the outputs of the classifiers from the learning process are used as inputs of the ensemble classifier that will then do the prediction. For the performance measures the most explored measures are: detection rate, sensivity, specificity, precision, and Receiver Operating Characteristics (ROC) value. The goal of this study was to compare all up-to-date ML ensemble algorithms and test their performance with inbalanced data. Baggedtree and GentleBoost classifiers show better performance.

Reference [33] proposes an ensemble-based classification model for network anomaly detection (EnClass). The model consists of two main phases. On the first phase, EnClass extracts important features from the dataset using hierarchical divisive clustering and Hoeffding bound. The features from the first phase are then optimized using Maximal Information Coefficient (MIC) and are used in the second phase to train the decision tree classifier. After this phases, the model is ready to detect anomalies in the network. This approach detects with high accuracy and low false alarms KDD'99 dataset anomalies, and it also detects some non-frequent attacks. System model is shown in 3.2.

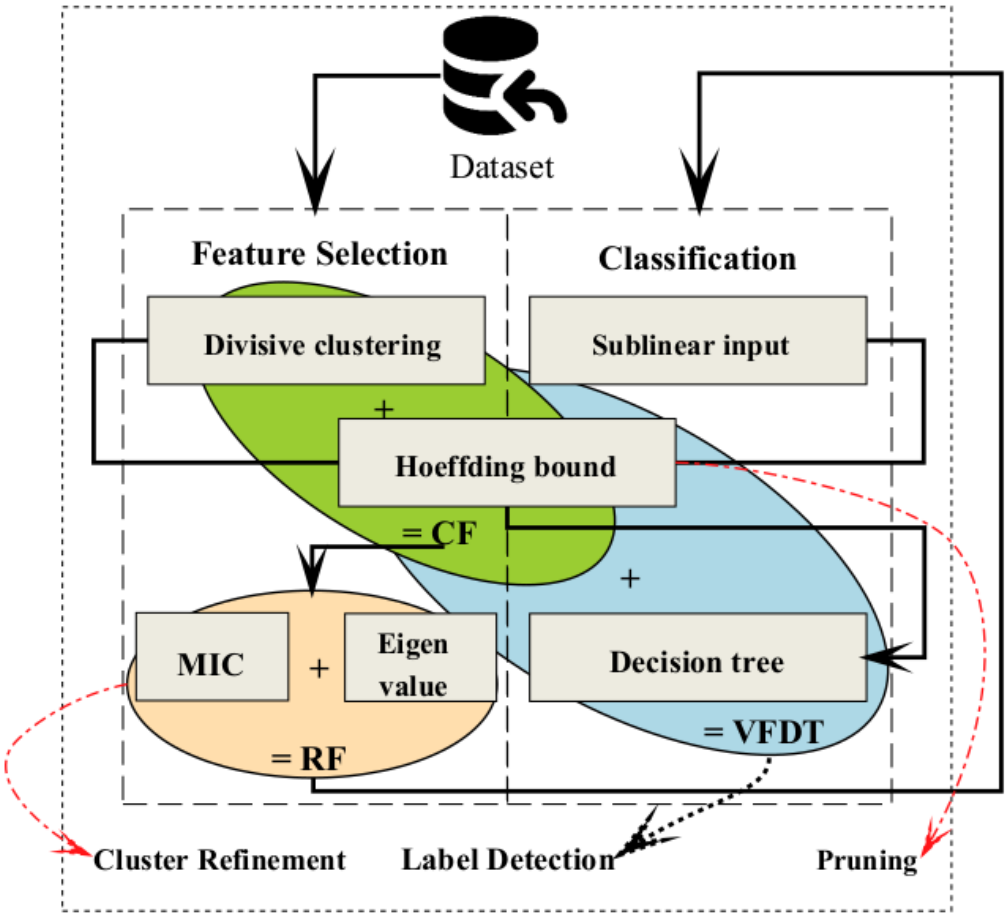

Figure 3.2: EnClass model [33]

3.5 Clustering and Outlier-Based

Clustering is the action of grouping data objects according to their similarity. Data objects within the same group (cluster) are more alike than those in other clusters. [12] To help understand the definition of clustering lets take a look into figure 3.3. Data objects are represented as points in the graphic and points in the same cluster are more similar then points in other clusters. Figure 3.3 a has examples of clusters as C1,C2,C3,C4,C5. Clusters are in form of an ellipse in order to have a better view of what a cluster is in the figure.

Outliers are data instances that do not conform with normal behavior. This means that outliers are considered anomalies. The detection of these outliers is simple as comparing the data instance to every other cluster and getting the similarity result. If the compared data instance does not belong to any cluster, it is considered as an outlier. In figure 3.3 b we can see outliers marked as letter O (O1, O2, O3, O4).

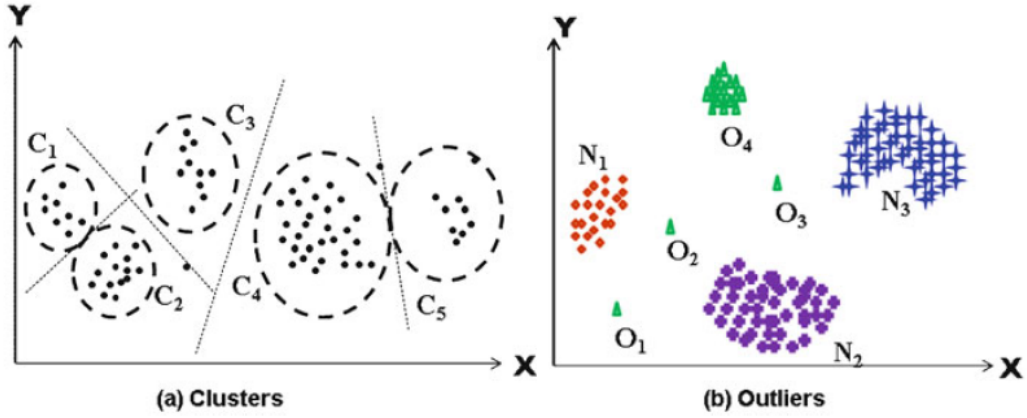

Figure 3.3: Clustering and Outliers in 2D where Clusters are represented as Cis and Outliers as Ois

Clustering techniques can be used in anomaly detection. Clustering study can be considered as an unsupervised method since it does not require any previous training. Unlabeled data will be divided into classes, which are clusters, and any data instance that does not belong to any cluster is considered an anomaly.

Reference [34] introduced Anomaly-based Data Mining for InTrusions (ADMIT), in which is an anomaly-based intrusion detection system that detects intruders by creating user profiles. It keeps track of the sequence of commands that users use on their computer. This sequence of commands will be used to create a user profile constituted by one or more clusters. Longest Common Subsequence (LCS) is used as the similarity metric and it also uses dynamic clustering algorithm that creates an initial set of clusters and then refines them. The detection works as follow: a user starts to type a sequence of commands; if it is a long sequence, it is divided into smaller sequences and then compared to the existing profiles; if the sequence does not belong to the normal user profile it is considered as an anomaly.

Reference [35] reported a study on distributed detection based on clustering. The Distributed Intrusion Detection System (DIDS) architecture is composed by Intrusion Detection Agent (IDA), which are placed in a distributed way in the network, and a central IDS. The first clustering happens on the IDA where it chooses the data candidates to be anomalies, the second one is on the central IDS where clustering tries to identify anomalies with the data received by IDA. At the IDA, the smallest clusters are considered to contain anomalies and will then be merged into a single cluster containing all the anomalies. After merging, the cluster containing anomalies is sent to the central IDS where it clusters again using a single-link hierarchical clustering algorithm. After central IDS clustering, the smallest k clusters are considered to contain true anomalies.

Reference [36] introduced a research on NIDS based on Improved K-means Clustering Algorithm. The author explains that when the original k-means cluster algorithm handles the value type, the selection of the initial cluster central point affects the algorithm. Also, when the data quantity is large, the time consumption of the computation distance is large accordingly. Consequently, a new algorithm is needed to improve the original algorithm dependence on initial cluster central point and time expenses. The new algorithm was built

with k-medoids algorithm, which replaces the mean value of the data object with the nearest point to the cluster. K-medoids cyclic method is proposed to solve the difference between the selection of the central point of the k-medoids algorithm and the fact that it is too large. This method uses dynamic central point renewal that helps to select k-medoids algorithm central point, triangle trilateral relations geometry theorem, that reduces the computation number of each iteration, computer expenses when data is large, and improve computation distance. This research shows that the improved algorithm improves detection rate and reduces false drop rates.

Reference [8] presents several outlier-based network anomaly identification techniques. Depending on the approaches used in outlier detection, outlier-based approaches can be classified as Distance-based, Density-based and Machine learning or Soft-computing-based. Distance-based methods are based on distance computing between data objects with clear geometric interpretation. Density-based methods estimate the density distribution of data. Low-density areas are considered anomalies and high-density areas as normal data. Soft-computing techniques are broadly used for intrusion detection because they can detect known and unknown attacks that have no precedent patterns by using their general capabilities.

Reference [37] developed a data distance measure to be used in outlier-based anomaly detection that contains a mix of categorical and continuous attributes. They defined an anomaly score that can be used to identify outliers in a mixed attribute space by considering dependencies within attributes of different types. Their anomaly score function is based on a global model of the data that is easily constructed by combining local models built independently at each node. Authors also show that their approach works with dynamic traffic. Reference [38] developed an improved clustering technique for outlier detection. The new algorithm is called Outlier Discovery and Clustering (ODC) and is based on the basic k-means algorithm. The first step of this algorithm is to assign each object to its closest centroid. Then it calculates the Sum if Squared Error (SSE)/Total Sum of Squares (SST) of the clustering in order to reduce errors; outliers are identified from the proposed definition of an outlier; when an outlier is detected, it is removed from the dataset and stored separately as an outlier; finally the centroids are recalculated. This process finishes when objects stop changing positions. Outliers are considered anomalies in network anomaly detection.

Reference [39] introduces a multi-step outlier-based technique that detects network-wide traffic anomalies. It consists of a feature selection technique, that efficiently identifies a relevant subset of features and tree-based subspace clustering algorithm that can be used in high dimensional datasets. They also brought in a fast distributed framework for the extraction and preparation of features in raw traffic data.

3.6 Knowledge-Based

Knowledge-Based techniques use previously gathered knowledge (rules, patterns) regarding abnormal events to match them with new events (network, host) in order to detect abnormalities. Expert systems, rule-based, and logic-based are some examples of knowledge-based methods, some will be discussed in this section.

An expert system is a technique that emulates the decision-making ability of a human expert. It is considered a rule-based system, that can have an associated knowledge base. It contains a rule engine that matches rules against system current state and depending on the matching results can trigger one or more rules.

Reference [40] proposes a rule-based anomaly detection on IP Flows that detects unwanted traffic using flow signatures. They use a machine learning algorithm named *Adaboost* to translate packet level signatures to work with flow-level statistics and associating packet level alarms with a feature vector that was derived from the same traffic flow. The authors use a set of rules similar to the ones used in SNORT (SNORT).

A well known rule-based intrusion detection and prevention system is SNORT[1]. SNORT is an open-source IDPS that matches incoming packets with a set of predefined rules. Rules are organized into priority classes based on the potential impact of alerts that match the rule. SNORT has been evolving for the last few years on which rules have their own documentation with potential false positives and negatives and also on what to do when the rule raises an alert.

Scalability is a problem if the number of packets increases, since packets will be matched against all rules and the number of rules can be quite large.

3.6.1 Artificial Neural Network Based Techniques

ANN are computer systems inspired by the recognition that the human brain computes in an entirely different level from the conventional digital computer. The human brain organizes its neurons to perform certain tasks way faster than any computer system. Such system acquires knowledge through a process of learning, which systematically changes interconnection strengths or synaptic weights of the network to achieve a certain objective. Reference [41] introduces a PCA-based neural network that can detect known as well as unknown attacks in real-time. In order to overcome the shortcomings of single-level structures (not detecting all attacks), the solution is based on a hierarchical intrusion detection model using PCA neural networks. Later on , reference [42], introduced a new approach for intrusion detection using PCA neural networks. The proposed system architecture is shown in 3.4. In their approach, the training stage is divided into two stages (layers). The first layer is composed of 5 PCA neural networks that are trained to cluster the input data according to the connection type. So, normal connections go to PCA-Normal, DoS connections go to PCA-DoS and so on. Each PCA neural network has 41 Processing Elements (PE) in the input layer, 5 PE in the hidden layer and 5 PE in the output layer. In stage 2 there is only 1 PCA neural network that uses as input stage 1 output and a little knowledge from the database. The author concludes that the proposed approach has better performance than some other PCA neural network approaches such as reference [41].

[1] `https://www.snort.org/`

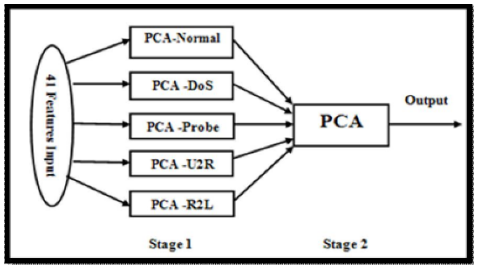

Figure 3.4: Proposed system architecture [42]

Reference [43] proposes an intelligent intrusion detection system based on soft computing techniques, namely neural networks and genetic algorithms, that improve intrusion detection performance. The proposed algorithm contains two main parts: the first being the initial neural network weights optimization using back propagation based genetic algorithm, and the second corresponding to the use of the the neural network to train intrusion detection data that will be used in the network model. This technique is said to improve intrusion detection performance.

3.7 Others

A NIDS includes one or more intrusion detection techniques and can have one or more subsystems associated. An example of a NIDS is Hierarchical Network Intrusion Detection System (HIDE) [44], which is a hierarchical anomaly network intrusion system that detects anomalies using statistical preprocessing and neural network classification. HIDE is a distributed hierarchical system, which contains various tiers and within each tier various IDA. IDA are part of an IDS and are used to monitor the activities of a host or a network. HIDE's IDA contains the following components: probe, event preprocessor, statistical processor, neural network classifier, and post-processor. The probe is used to collect network traffic, abstract the traffic into a set of statistical variables that reflects network status, and periodically generating reports for the event processor. Event Preprocessor receives reports from both probe and low tier IDA's and converts the information into the format required by the statistical model. Statistical Processor keeps a reference model of the typical network activities, compares the reports from the event preprocessor to the reference models and forms a stimulus vector to feed into the neural network classifiers. Neural Network Classifier analyzes the stimulus vector from the statistical model for abnormal traffic. Post Processor is used for generating reports for high tier agents.

CHAPTER 4

Implementation

4.1 File sharing services

OwnCloud [1] is a file sharing service that offers enterprises file sharing capabilities which can be used in the client's facilities or hosted in a cloud server. Being hosted in client facilities, it uses clients storage and is managed by clients IT. This service offers features like file firewall, antivirus scanning, logging, authentication, collaborative editing of office documents and much more. It also has a desktop application that is used to automatically sync files between server and user. OwnCloud runs over HTTP/HTTPS, which uses TCP ports 80/443.

NextCloud [2] offers file sync and share solutions that keep the client data into client control. It is a self-hosted solution, open source and designed to be easy-to-use. Similar to OwnCloud, nextcloud uses WebDAV technology that uses HTTP/HTTPS and works under TCP ports 80/443.

Dropbox [3] is probably one of the most used file storage and sharing service with more than 500 million registered users. Dropbox provides a simple and easy-to-use application, supporting almost every Operating System (OS) letting its users sync and access their files from nearly everywhere. There are some variants when sharing a folder. The user can make a folder public by copying it to the "Public" folder that Dropbox automatically creates and just send the public link to the person that the user wants to share the folder with. The user can also share a folder with a group of people by inviting them via email so they can access the shared folder. The invited people will have to create, edit, and delete permissions that can be changed by the person who created the shared folder. The person receiving the email will need to have a Dropbox account but does not need to install the app to get access to the folder. Working in group is easy as Dropbox support collaborative file sharing, which means that one or more users can edit the same file at the same time. This can be usefull when working in group.

[1] https://owncloud.org/
[2] https://nextcloud.com/
[3] https://www.dropbox.com/

Syncing is rather simple, the user just needs to install the Dropbox desktop app and then wait for the files to sync automatically. This syncing feature can be turned off.
For an additional layer of security, Dropbox offers two-factor authentication. To secure file sharing between dropbox applications and their servers Dropbox uses SSL/TLS which uses Advanced Encryption Standard (AES). The application layer used is HTTPS that uses port 443.

IDrive [4] offers a free 5GB account, a 2TB personal tier for $69.50 a year and also a business-level subscription that adds sub-account management features, priority support, server backup, and an unlimited number of users per account for $99.50 a year. IDrive offers an easy setup, unlimited devices per account, disk image backup, file explorer integration, file syncing. When creating an account the user has the possibility of choosing between a private encryption key or the default IDrive-managed key that will be used to encrypt the files. IDrive has an interesting feature that lets their users restore their files into some folder using checkboxes or just restore everything into a place. This feature also lets the user to choose between all 10 different file versions that are stored in the server (there is an SOS version that has unlimited versioning). Within this feature, there is a new feature called Snapshots that aims to address ransomware concerns. It shows a historical timeline of the user backup with the possibility of recovering files at any point along the timeline. As any other storage and file sharing service, IDrive also offers applications for almost every OS, which let their users access, sync, backup, and restore files. IDrive also uses HTTPS for server-client communications running over TCP port 443. [5]

SugarSync [6] offers easy setup and intuitive usage, but it lacks capabilities such as collaborative editing and two-factor authentication. Just like other file-sharing services, SugarSync lets their users share files and folders to anyone that has the link or to a specific person or group of people. There is no online collaborative tool, so the users cannot edit the same file simultaneously. Overall, SugarSync is easy to use, has a simple interface, and offers all file sharing basics. Regarding security, it uses TLS encryption, so in a technical view information is sent via HTTPS that runs over TCP port 443. [7]

OneDrive [8] is a file-sharing and storage service provided by Microsoft that has amazing features such as music streaming, shared desktop-folder syncing, real-time co-authoring in office, notifications of other users editing, and full-document searching in shared files. Onedrive file and folder syncing is similar to Dropbox. Onedrive becomes more of a built-in capability when using Windows 10. In Windows 10 the user can choose which OneDrive folders are synced or if every folder is synced. Onedrive works better in windows 10, although it can be used in other OS. In terms of security, OneDrive offers two-step authentication mechanism, advanced security in case the user subscribes to office 365 and on the communication level it offers SSL/TLS encryption. Data between client and server are done via HTTPS, which

[4] `https://www.idrive.com/`
[5] `https://www.pcmag.com/article2/0,2817,2362675,00.asp`, accessed 29/10/2018
[6] `https://www2.sugarsync.com/`
[7] `https://www.pcmag.com/article2/0,2817,2343598,00.asp`, accessed 29/10/2018
[8] `https://onedrive.live.com/about/en-us/`

works under TCP port 443. [9]

Filecloud [10] is a file-sharing solution that offers both online, files are stored in Filecloud server, and own server where files are stored within enterprise servers. It has a large set of features for both server and online solutions. It has public and private shares, microsoft office online integration, unlimited file versioning (good feature in case of ransomware), comments on files and folders and much more. On the security level, has the possibility of using two-factor authentication, data is encrypted with 256-bit and is sent over SSL, supports Security Assertion Markup Language (SAML) and it also ha anti-ransomware and anti-virus scanning.

After studying file-sharing services above, we can see that most of them use SSL/TLS for encrypting data, HTTPS application layer protocol on TCP port 443 as the transport layer.

For an "offline" file-sharing service we have OwnCloud (free) or Filecloud, which offers the possibility of implementing their solution in any machine, meaning that an enterprise can implement it on their servers and host a file-sharing service. We are going to use OwnCloud as the internal file-sharing server for testing purposes.

4.1.1 Vulnerabilities And Attacks

In chapter 2 we talked about licit and ilicit service usage, types of attacks (ransomware and theft), source of the attack (external or internal) and attack propagation. Now we will talk about file-sharing vulnerabitilies and types of attacks to file-sharing services and how they have been handled.

A vulnerability is a weakness that can be exploited by an attacker. This weaknesses can be something like an open port on a computer. If there is a point of vulnerabitily, there is a chance that someone will exploit it and use it to perform unauthorized actions.

Since file-sharing services are hosted on a server, there are many entry points that can be exploited by an attacker. Therefore, there are many attacks that can be exploited by an attacker.

Services in general are normally contained in a Virtual Machine (VM) or in Docker (more lightweight) and hosted on a HTTP server software such as Apache. The software contained within the service container might contain vulnerabilities that can be exploited by the attacker to get access to the service. Once the attacker gets access to the server, it can spread a malware to the users using the service and therefore propagate to other machines.

Reference [45] was able to identify five categories of Docker vulnerabilities: insecure production system configuration; vulnerabilities in the image distributon, verification, decompression, storage process; vulnerabilities inside the images; vulnerabilties directly linked to Docker or libcontainer; vulnerabilities of the linux kernel (Docker software is only possible in linux systems). By studying these vulnerabilities they showed that it is possible to insert malicious code into production.

After getting access to the container, the attacker can further exploit any vulnerabilities on the HTTP server and get access to the service.

[9] `https://www.pcmag.com/article2/0,2817,2409569,00.asp`, accessed 29/10/2018

[10] `https://www.getfilecloud.com/`

4.2 Network Profiling

The creation of profiles in networking is useful for differentiation and anomaly detection. Machines normally have different behavior in a network, some produce a lot of traffic, some produce less, some produce traffic in a way that probably no other machine produces. But, in some cases, it can happen that a group of machines have similar behavior in a network and in this case, a single profile corresponding to the group of machine's behavior is used. To help with anomaly detection, we create a profile for each machine in the network and compare all the new traffic from the machine to the profile created in the training stage, in order to detect anomalies. To have a clear look in what a profile in the network actually is, we have to define what an observation window is.

An observation window is composed by multiple samplings/counting metrics that quantify activity events such as the start/end of an activity (traffic flows, service usage,..), amount of activity (activity duration, actions per sampling interval, ...) and activity targets (number of IP addresses contacted, UDP/TCP used ports, services, ...). An observation window can also be sequential or sliding. In a sequential observation window, the windows have a fixed size and the decision interval is equal to the window size. In a sliding window, although the window size is also fixed, a window slide is done in order to get longer periods of observations while maintaining a short period of decision. The sliding window is helpful when we have small samples.

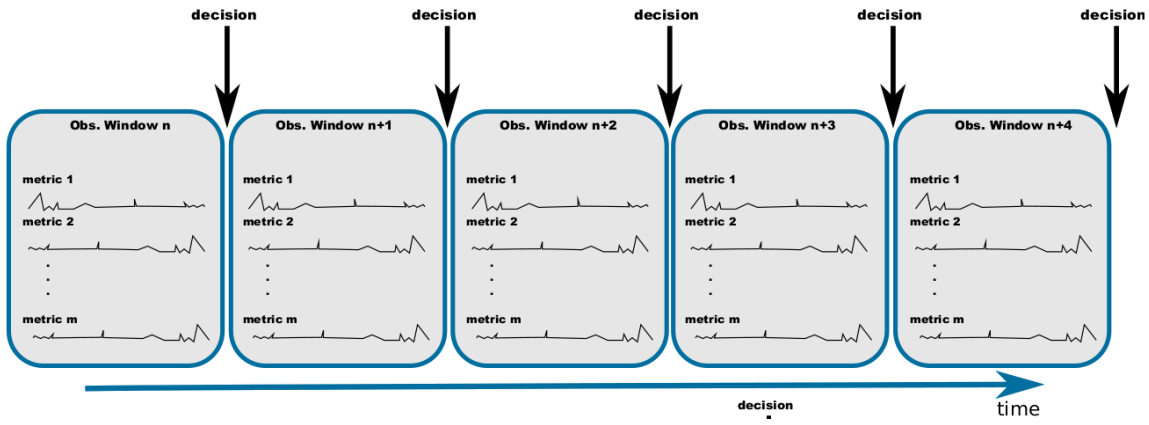

Figure 4.1: Sequential window example "DDR-Network Monitoring 28/03/2018"

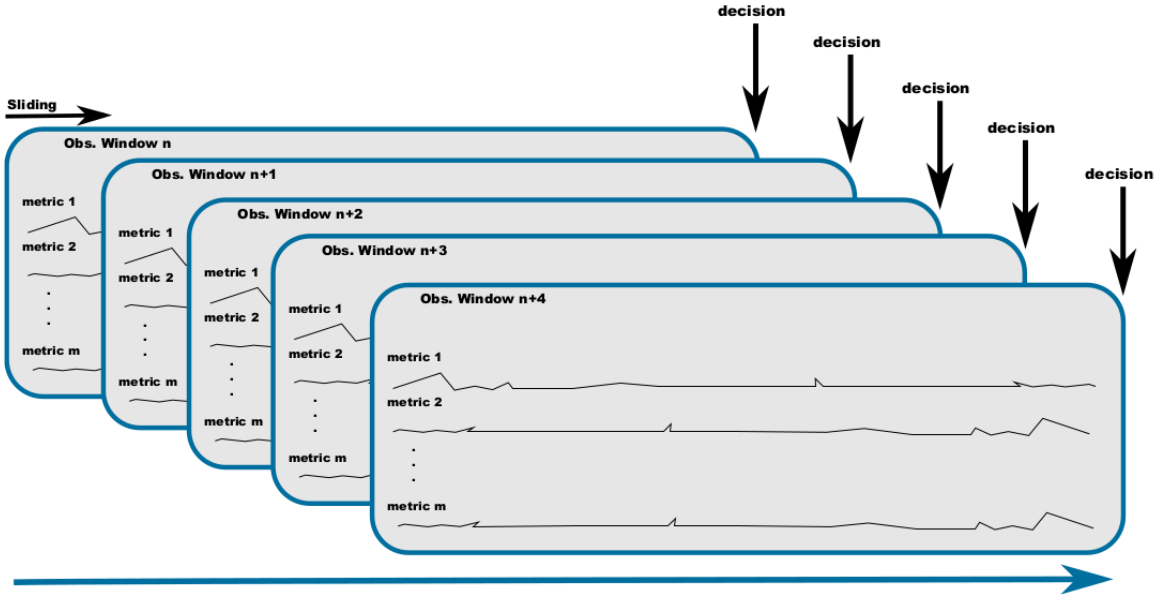

Figure 4.2: Sliding window example "DDR-Network Monitoring 28/03/2018"

An event is normally described by multiple metrics/features (mean, variance, median, ..). In our approach we will have two different sizes for the observation windows, delta T1 window has 1 second size, and delta T2 window (event window) has 120 seconds size. In deltaT1 window we do the counting of different IP, uploads, downloads, specific ports. In deltaT2 we apply statistics to the counters.

Now that we have the profile of a machine or group of machines (composed by a group of event windows/observation windows) we need to make the decision. To make a decision we can take a statistical pattern or ML. In our approach, we use ML since it can learn through time and make better decisions. First, we train the machine with the training dataset (all the profiles) and then compare it to all the new event windows and decide if the new point is part of the normal or anomaly group.

4.3 Our Appoach

The goal of this thesis is the detection of anomalies in network file-sharing. To detect the anomalies we divided our architecture into four main modules: Sniffer, Data pre-processing, Analyser and Decision-maker. The Sniffing module reads all packets from the network and periodically saves a temporary pcap file. Data pre-processing takes that pcap file and processes data to be used in the Analyser module. The analyzer will take the pre-processed data and use it as input data for our ML approach, that will then output if there was an anomaly or not and make a decision. We divided our Analyser module into three distinct agents, where each agent has the goal of detecting one of the anomalies (propagation, ransomware, theft). A simple architecture is shown below:

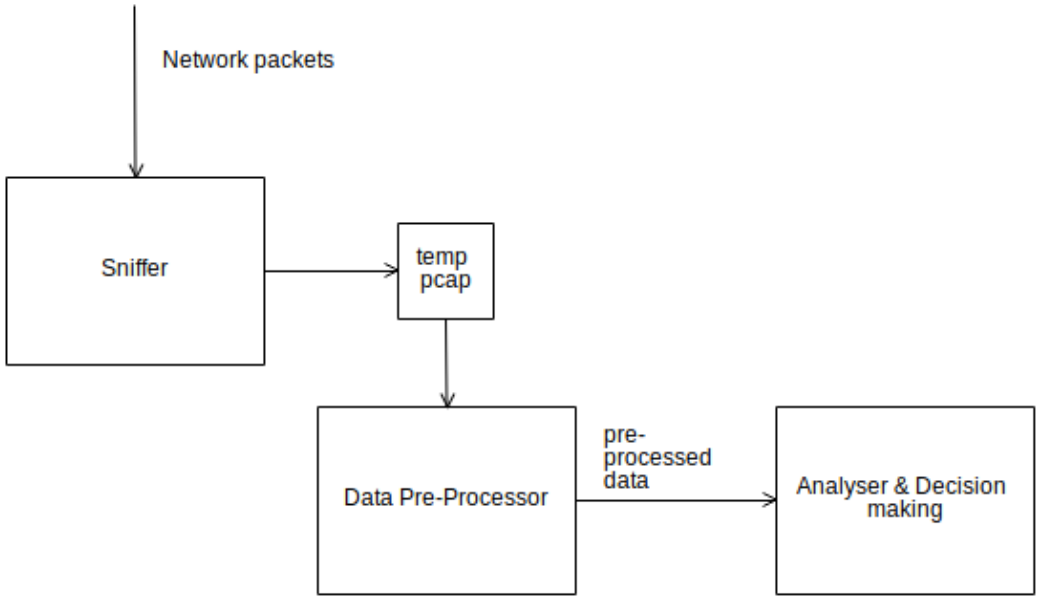

Figure 4.3: Simple model architecture

4.3.1 Dataset preparations

In order to create our dataset, we needed to capture both normal and anomaly behavior. For normal behavior, we have Youtube, Netflix, Browsing, Owncloud, Dropbox. For anomaly behavior, we have propagation using the Eternalblue exploit (uses SMB), ransomware in both Owncloud and Dropbox servers, and theft using both SFTP and Dropbox external servers.

All behaviors were captured using the Wireshark [11] software. For Youtube and Netflix we simply choose a video and let the autoplay on for about one hour. For Browsing, we tried to be as normal as possible by surfing the web. Now for both normal and anomaly behavior in Dropbox and Owncloud we needed scripts to simulate the normal human behavior and ransomware.

First, we needed a folder to do the file operations on, so we created a folder with four main folders. Each folder has a random value of files from a given range and each file of each main folder has a size value from a given normal distribution. Main folder 1 is named 'dir1', main folder 2 is named 'dir2' and so on. Each main folder has twelve other folders and in each one of the twelve, three other folders. Table 4.1 shows all values used in each one of the four main folders. In Table 4.2 shows our main folder that it is used for file sharing testings.

Now that we have our testing folder ready, we need to create human-like behavior while doing file sharing in both Dropbox and Owncloud servers. As referenced in 2.6 users have different time between actions while doing file operations. To achieve a human-like behavior we decided to get the time between actions value from an exponential distribution with a certain mean value. We decided to use three different mean values which will be our three different user profiles. Profile 1 with a mean value of 20 seconds, profile 2 with 120 seconds

[11] `https://www.wireshark.org/`

Table 4.1: Directory and files specifications

Directory	mean	standard deviation	lower limit	upper limit	Range of files in each directory
dir1	10Kb	5Kb	5Kb	15Kb	1-13
dir2	100Kb	50Kb	50Kb	150Kb	1-8
dir3	1Mb	0.5Mb	0.5Mb	1.5Mb	1-8
dir4	10Mb	5Mb	5Mb	15Mb	1-6

Table 4.2: Test folder and specifications

Directory	Total number of files	Total size
dir1	282	2.7Mb
dir2	199	20.8Mb
dir3	184	182.9Mb
dir4	145	1.4Gb
main dir	810	1.6 Gb

and profile 3 with 300 seconds. The different file operations are create, edit and delete. Since edit is the most common operation it has a weight of 90% in each user operation, create and delete have both 5%. We created a python script that, given the profile (20,120,300 mean values in seconds), do file operations and then sleeps for a certain amount of time, depending on the value outputted from the exponential distribution variable, meaning that we have three different profiles for each of the servers Dropbox and Owncloud.

For ransomware, the folder to encrypt is our main folder referenced above. We made a script that simply encrypts the files with one second delay between files, changing the file name to 'filename.enc' and increasing its size a little. For Theft, we also used the main folder with no alterations.

Now that we have all captures needed, in .pcap or .pcapng, we need to extract all relevant features and apply statistics on them to, later on, use as input for our machine learning approach. We read captures using pyshark library and python as the programming language. After getting all packet information needed, we proceed to do the countings for the features of interest. Selected features are shown in 4.3.

Features 'diff_ip' and 'diff_port' are important as in some case scenarios connecting to some IP in a different port is considered an anomaly. 'smb' is an important feature for the scenario 4.4.1 where we detect malware propagation and an example of a no sync server ransomware and theft. Feature 'arp' is used for Address Resolution Protocol (ARP) counting which is important for the detection of malware propagation. 'Up' and 'down' are also important features when detecting the general use of ransomware and theft where the number of uploads and downloads in a time window is an important factor. 'Http' and 'https' were picked as almost every file sharing service use those protocols. 'Ext' is used to count external IP addresses contacted, it is important for theft anomaly detection. The feature 'silence' is good to catch robotic like behavior like in ransomware case where it monotonously encrypts

Table 4.3: Selected features and description

Feature	Description
arp	Counter for ARP packet type
ip	Counter for IP packet type
tcp	Counter for TCP packet type
udp	Counter for UDP packet type
icmp	Counter for ICMP packet type
pkt_count	Number of packets
diff_ip	Counter for the different IP addresses on the observation window
diff_port	Counter for the different ports on the observation window
smb	Counter for SMB packet type (tcp ports: 445, 139, 137)
pkt_length	Sum of packet lengths
up	Counter of uploads
down	Counter of downloads
https	Counter of HTTPS protocol (tcp port: 443)
http	Counter for HTTP protocol (tcp port: 80)
ext	Counter of the number of external IP's contacted
fin	Counter of the number of FIN flags
syn	Counter of the number of SYN flags
rst	Counter of the number of RESET flags
psh	Counter of the number of PUSH flags
silence	If the number of packets in t1 (1 second) window is below 3, silence is 1

all files. Finally 'arp', 'ip', 'tcp', 'udp', 'icmp' are counters used for generalization in case a zero-day attack.

After applying countings to the features, we apply statistics to each feature. The statistics are mean, median, variance, percentile (with 75,90 and 95). For the 'silence' feature we only apply mean and variance. Then, we write each observation window statistics into a file with the respective label (0- normal, 1 - anomaly) where the columns are the features with the applied statistics and the lines the number of DeltaT2 observations, each line represents 120 seconds of observations. We also added a label for each application for better result reading and to detect which applications are better detected and which are not. Labels are shown in table 4.4.

Table 4.4: Dataset labels and their description

Label	Capture description	Dataset
1	Normal browsing	main
2	Youtube	
3	Normal Dropbox file sharing capture with time between actions (ta) of 20	
4	Normal Dropbox file sharing capture with ta = 120	
5	Normal Dropbox file sharing capture with ta = 300	
6	Normal Owncloud file sharing capture with ta = 20	
7	Normal Owncloud file sharing capture with ta = 120	
8	Normal Owncloud file sharing capture with ta = 300	
9	Owncloud anomaly (ransomware)	
10	Dropbox anomaly (ransomware)	
11	Netflix streaming	new
12	SMB propagation anomaly	main
13	SMB propagation anomaly 2	new
14	Dropbox anomaly 2 (ransomware)	
15	Dropbox anomaly 2 (theft)	
16	Dropbox theft	main

4.3.2 Supervised Machine Learning preparations

In section 2.15 we talked about feature reduction and its importance. Now that we have our dataset set ready, we can proceed to do data normalization and reduction. Normalization is important because ML estimators perform better and it also helps PCA choosing the best axis. For normalization we use sklearn StandardScaler [12], we fitted the training features and transformed both training and test.

Now that we have our dataset normalized, we use PCA to reduce our features while achieving a variance of 80% or better. For that we use sklearn PCA [13]. If we have an "n_component" between 0.80 and 1, and an "svd" equal to "full", the method will try to output the best axis where the variance meets the required values (80% to 100%). After this step, we have our dataset ready to do ML testings.

As said in 4.3, we divided our analyzer module into detecting all three anomalies separately. In order to detect each of the anomalies separately, we divided our dataset into propagation, ransomware and theft datasets. In order to not confuse the anomalies with their datasets, we will be calling propagation dataset "pdataset", ransomware dataset "rdataset" and theft

[12] `http://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.StandardScaler.html`, last accessed 19/10/2018

[13] `http://scikit-learn.org/stable/modules/generated/sklearn.decomposition.PCA.html`, last accessed 19/10/2018

dataset "tdataset". For all datasets, we included all normal data (youtube, browsing, ...). For pdataset we also added propagation anomaly behavior (label 12, 13 in table 4.4), for rdataset we added ransomware behavior (label 9, 10, 14 from table 4.4) and for tdataset we added theft anomaly behavior (label 15, 16 from table 4.4).

Now in 4.4 we will address practical scenarios .

4.4 Pratical Scenarios

In this section we present the different practical scenarios that will be studied in this thesis. We will detect two anomalies, ransomware, theft, and how they both propagate. There are several scenarios in an enterprise network but we will be addressing three of them.

4.4.1 Scenario 1

In scenario 1, we will be focusing on the detection of malware propagation. In this scenario we assume that the machine is already infected with a malware and our goal is to detect its propagation to other machines. To simulate a malware propagation we are going to use the EternalBlue exploit, which was an exploit to a vulnerability on SMBv1 that WannaCry used in order to propagate to other machines. We are going to use a script with some changes: number of IP's that the script tries to establish a connection with (range of IP's); sleeping time between exploits.

For this scenario capture (label 12 in table 4.4) we let the script run with a range of IP of 10.2.128.6 to 10.2.128.50 with a random time between exploits of a range between 1 and 3. For the second anomaly capture (label 13 in table 4.4) we used a range of 10.2.128.6 to 10.2.128.100 with a range between exploits of range 3 and 10.

We know that in order to propagate, the script sends ARP requests to the victims IP addresses in order to verify if the machine with the IP exists and is running. So, having an ARP protocol counter on our feature set is important. Besides, SMB protocol uses TCP port 445, which is also going to be on the feature set. We will also look into IP, TCP, UDP, different ports and IP's, upload and downloads counters for generalization.

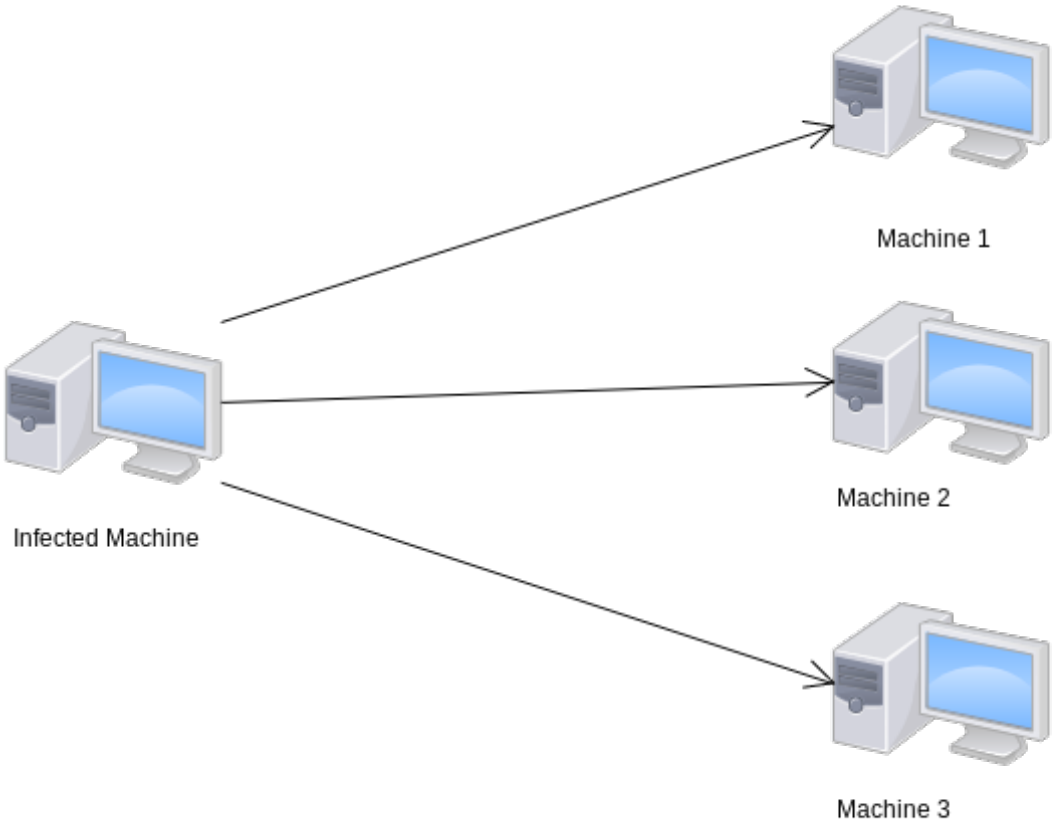

Figure 4.4: Scenario 1

4.4.2 Scenario 2

In scenario 2, we use an internal server, in our case owncloud, with file sharing capabilities. The goal is to detect both ransomware and theft attacks while capturing packets when the internal server file sharing service is being used. In this scenario, we assume that the server has a client with sync capabilities, which means that all files are synced in the user computer if the user has the file sharing service client installed and the network is working. With this, we assume that the files that are on the user machine are exactly the same as the files that are on the internal server.

In case of theft, since the files are all synced there is no need for any requests to the internal server, so there will not be any or little downloads from the server into the infected machine. But the number of uploads from the infected machine will be high since its goal is to send the information somewhere else. We also assume that the goal of the attacker is to send the data to an external server from the internal enterprise network, so in this case, we also have different IP addresses and more external IP addresses.

For this scenario, we captured ransomware and theft behavior. For ransomware (label 9 in table 4.4) we used a script that given a directory, encrypts all data with one second time sleep between files and replaces the old file with the encrypted one, changing its name to 'filename.enc' and increasing file size by a little. In this case features like packet length, the

number of TCP packets and silence between actions, are relevant to detect ransomware. The fact that there is a time sleep of one second, makes the encryption method monotonous and robotic. 'Silence' feature is going to have great importance when detecting ransomware. For the capture of theft behavior, we uploaded a directory into an external IP server making it seem like someone was sending files to an unknown external server.

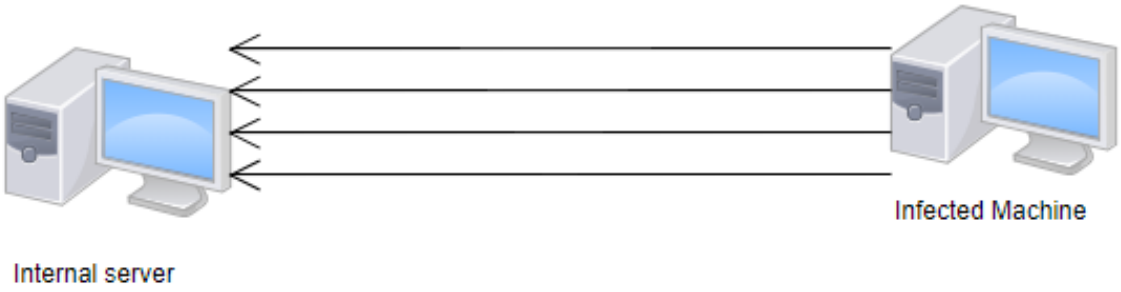

Figure 4.5: Scenario 2 ransomware

4.4.3 Scenario 3

In this scenario, we will simulate file sharing through the cloud. We will be using an external server (example: Dropbox), in which the user machine communicate through the file sharing service API. In this case, the server is also synced with the machines, so it is similar to scenario 2. The main difference is that in this case, we will be communicating with an external server which has an external IP and not an internal IP as in scenario 2.

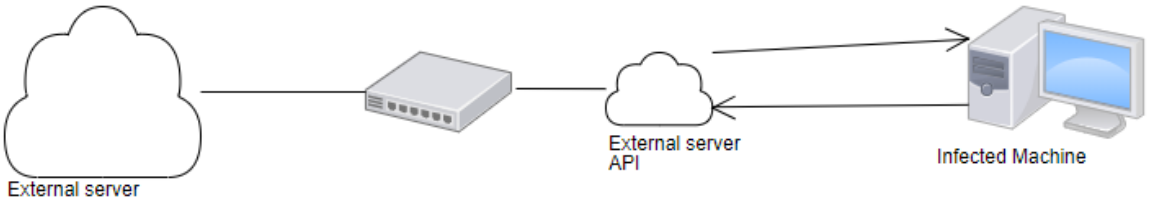

Figure 4.6: Scenario 3

4.5 Placement of IDS Sensors

The placement of an intrusion detection sensor is of great importance because we have to choose a place where we can get all packets that are passing through the network. So a common location is just after the network firewall like shown in figure 4.7.

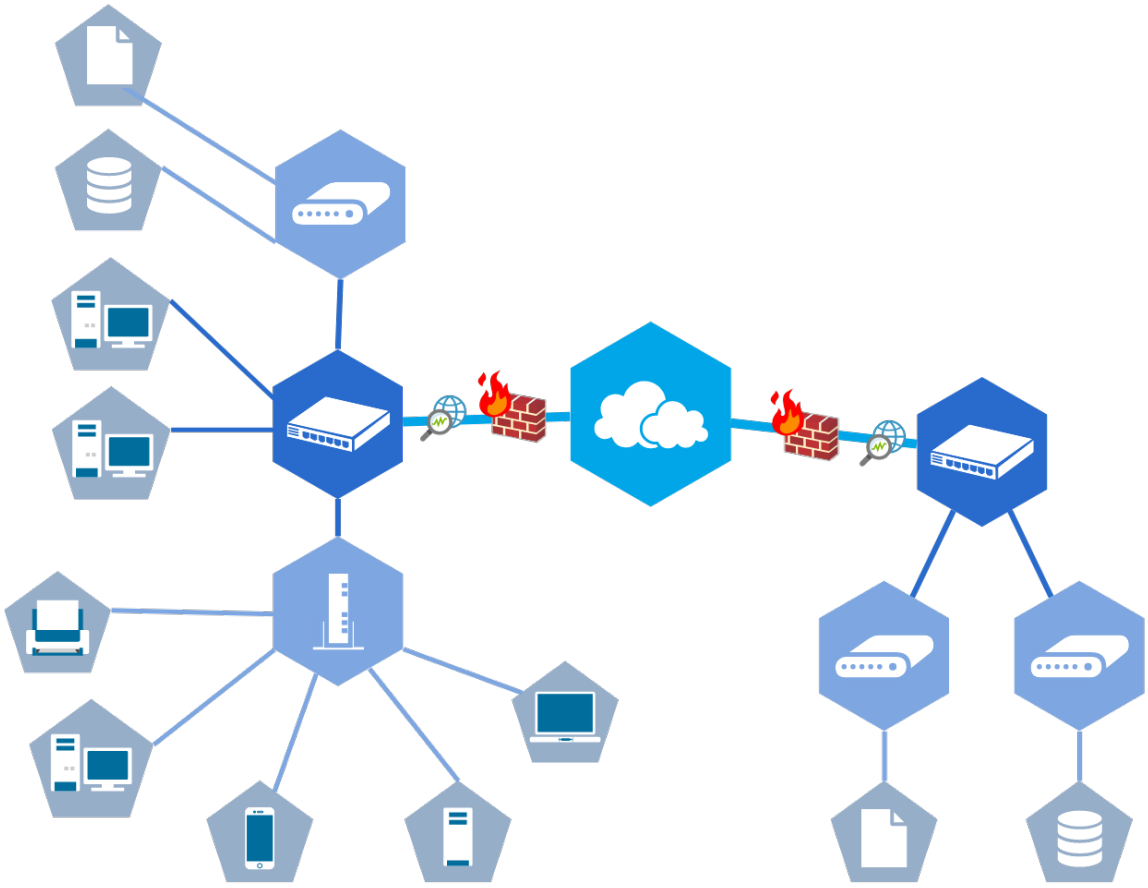

Figure 4.7: One example of an IDS sensor

CHAPTER 5

Experiments

5.1 Anomaly Detection using classification

The objective of the classification approach is to detect anomalies by classification. In this approach, we classify all normal behavior into normal class and all anomaly behavior into anomaly class. The anomalies we want to detect are ransomware, theft, and their propagation. This is done by studying network profiles from different machines.

Now that we have our dataset for training and testing, with both normal and anomaly classes (label 0 - normal, label 1 - anomaly), the next step is to test the dataset with different algorithms and parameters in order to get the ML algorithm that detects our anomalies in the different scenarios with little false positives and high accuracy score. In order to validate our model, we implemented a cross-validation approach. For each tested algorithm we first shuffle all the dataset, split data into 80% train and 20% test. The training dataset is used to fit the model of the chosen algorithm and the test to use as predictive data. This process is done k times. We keep each iteration predictions in order to test the overall model accuracy and so validate our model.

For further model evaluation, we created a new dataset with data that was not seen in the main dataset. This new data is not part of the main dataset, meaning that it is not used in the shuffle or train. For this new data, we use the trained model from the main dataset and then predict using the new data. With this, we can have a better model evaluation. In all datasets (propagation, ransomware, theft) we have Youtube as being the normal video behavior, so in order to test a similar normal behavior, we use Netflix as the new normal video behavior in the new dataset. For propagation, we have the anomaly in the main dataset and we have another propagation anomaly in the new dataset. This is also true for ransomware and theft, we have for each, a new anomaly in the new dataset for better model evaluation.

For testing, we will be using scikit-learn [1] libraries that have ML algorithms implementations. We decided to use sliding windows because it allows us to have more observations in each delta t2 window. For testing, we used a slide of t = 10 seconds in increments of 10

[1] `http://scikit-learn.org`, last accessed 05/12/2018

until 120. A slide t of 10 seconds means that window X has 110 seconds of the window X-1, and only 10 seconds of new observations. App true class is the label value of the application we are testing, label 0 means normal behavior and label 1 means an anomalous behavior. Right prediction means that the prediction of the application label was right, and the wrong prediction means that it was wrongly predicted. A right prediction would be 0 for Browsing and a wrong prediction would be 1 for Browsing. Testing and results will be shown in the next subsections.

5.1.1 SVM

As we already know, SVM use supervised learning methods that are used for classification, regression and outlier detection. SVM also offers different kernel functions for the decision function. We tested our dataset on C-Support Vector Classification (SVC) with Radial Basis Function (RBF) kernel, which is a classification approach.

When training SVC we must consider penalty and kernel coefficient parameters. In the scikit-learn library, C is the penalty parameter and gamma the kernel coefficient. C parameter is common to all kernels and trades off misclassification of the training examples against the simplicity of the decision. Low C makes the decision smoother, while a high C goal is to classify all training examples correctly. While C is used for all kernels, gamma is used in nonlinear hyperplanes. Gamma defines how much influence the current dataset has. The higher the gamma value, the more it tries to fit the training dataset. In RBF, we have to take C and gamma values into consideration.

We tested the machine with its default values. Results for each anomaly detection are shown below.

Propagation

Table 5.1 shows the result metrics and number of right/wrong predicted classes for the best two slides of propagation detection using SVC algorithm.

Table 5.1: Table with SVM results for propagation detection

Window slide t	Application	App true class	right prediction	wrong prediction	Accuracy %	F1 %	Precision %	Recall %	PCA variance %
20	Browsing	0	375	0	98.37	83.21	100	71.76	99
	Youtube	0	356	0					
	Dropbox 20	0	306	0					
	Dropbox 120	0	354	0					
	Dropbox 300	0	336	0					
	Owncloud 20	0	351	0					
	Owncloud 120	0	346	0					
	Owncloud 300	0	356	0					
	Propagation Anomaly	1	122	48					
	Netflix	0	2270	0	90.26	54.67	100	37.62	
	Propagation Anomaly new test	1	158	262					
40	Browsing	0	180	0	97.03	64.39	100	51.11	99
	Youtube	0	164	0					
	Dropbox 20	0	174	0					
	Dropbox 120	0	175	0					
	Dropbox 300	0	182	0					
	Owncloud 20	0	180	0					
	Owncloud 120	0	168	0					
	Owncloud 300	0	167	0					
	Propagation Anomaly	1	46	44					
	Netflix	0	1140	0	90.60	56.20	100	39.52	
	Propagation Anomaly new test	1	83	127					

Ransomware

Table 5.2 shows the result metrics and number of right/wrong predicted classes for the best two slides of ransomware detection using SVC algorithm.

Table 5.2: Table with SVM results for ransomware detection

Window slide t	Application	App true class	right prediction	wrong prediction	Accuracy %	F1 %	Precision %	Recall %	PCA variance %
120	Browsing	0	65	0	94.81	61.90	100	46	95
	Youtube	0	56	6					
	Dropbox 20	0	59	0					
	Dropbox 120	0	57	0					
	Dropbox 300	0	55	0					
	Owncloud 20	0	64	0					
	Owncloud 120	0	58	0					
	Owncloud 300	0	56	0					
	Owncloud Ransomware	1	14	12					
	Dropbox Ransomware	1	9	15					
	Netflix	0	380	0	97.44	87.5	100	78	
	Dropbox Ransomware new test	1	39	11					
50	Browsing	0	137	1	96.37	75.98	100	62.5	95
	Youtube	0	144	0					
	Dropbox 20	0	133	0					
	Dropbox 120	0	140	0					
	Dropbox 300	0	135	0					
	Owncloud 20	0	133	0					
	Owncloud 120	0	145	0					
	Owncloud 300	0	153	0					
	Owncloud Ransomware	1	46	14					
	Dropbox Ransomware	1	29	31					
	Netflix	0	902	8	98.16	92.26	94.36	90.83	
	Dropbox Ransomware New test	1	109	11					

Theft

Table 5.3 shows the result metrics and number of right/wrong predicted classes for the best two slides of theft detection using SVC algorithm.

Table 5.3: Table with SVM results for Theft detection

Window slide t	Application	App true class	right prediction	wrong prediction	Accuracy %	F1 %	Precision %	Recall %	PCA variance %
10	Browsing	0	686	0	99.95	94	100	89.99	80
	Youtube	0	668	0					
	Dropbox 20	0	678	0					
	Dropbox 120	0	709	0					
	Dropbox 300	0	697	0					
	Owncloud 20	0	693	0					
	Owncloud 120	0	691	0					
	Owncloud 300	0	728	0					
	Dropbox theft	1	27	3					
	Netflix	0	4540	0	99.56	40	50	33.3	
	Dropbox Theft new test	1	10	20					
20	Browsing	0	345	0	99.57	46.67	60	40	80
	Youtube	0	341	0					
	Dropbox 20	0	345	0					
	Dropbox 120	0	362	0					
	Dropbox 300	0	356	0					
	Owncloud 20	0	331	0					
	Owncloud 120	0	367	0					
	Owncloud 300	0	333	0					
	Dropbox Theft	1	8	12					
	Netflix	0	2270	0	99.39	40	60	30	
	Dropbox Theft New test	1	6	14					

SVC algorithm conclusions

As we can see from the tables 5.1, 5.2 and 5.3 the accuracy scores of SVC algorithm are pretty good but there is a need of wrong class predictions reduction. Wrong prediction class means that the model didn't predict the application true class. Let's say, for example, propagation anomaly, should be predicted as one since the class for the anomaly is one, but our trained model predicted 48 wrong classes in slide t=20 on our main dataset, meaning that it predicted 48 zeros where it should be ones. We have cases where anomaly behavior is considered normal and normal is considered as an anomaly. The goal is to reduce the wrong predicted classes.

We will now be testing the same datasets in Decision Tree (DT) algorithm in section 5.1.2.

5.1.2 Decision Trees

DT are non-parametric supervised learning methods used for classification and regression. The goal is to create a model that can predict the values of a target variable by learning simple decision rules inferred from the dataset features. In DT the deeper the tree, the more complex the decision rules and the fitter the model.

We will be using scikit-learn terminologies but the meaning will be the same for any implementation, the name of the parameter might be different from other implementations.

To set up the tree we have:

- criterion : Which is the function that measures the quality of a split. Supported values on sklearn: gini (for Gini imputiry) and entropy (for information gain).
- splitter : Is the strategy used for the splitting of each node. Supported values on sklearn: best (for the best split) and random (for a random split).
- `max_depth` : Defines the maximum depth of a tree. If no value is given, the tree will expand until all leaves are pure or until all leaves contain less than `min_samples_split` samples. The deeper the tree, the more splits it has and the more information it has about the data.
- `min_samples_split` : Is the minimum number of samples required to split a node.
- `min_samples_leaf` : Is the minimum number of samples required to be a leaf node.
- `max_features` : Is the number of features to consider while looking for the best split.
- `max_leaf_nodes` : Defines the max number of leaf nodes in the tree.
- `min_impurity_decrease` : A node will be split if this split induces a decrease of impurity greater than or equal to this value.

We tested the model with default values for each anomaly, the results are shown below.

Propagation

Table 5.4 shows the result metrics and number of right/wrong predicted classes for the best two slides of propagation detection using DT algorithm.

Table 5.4: Table with DecisionTreeClassifier results for propagation detection

Window slide t	Application	App true class	right prediction	wrong prediction	Accuracy %	F1 %	Precision %	Recall %	PCA variance %
90	Browsing	0	89	0	98.06	82.80	86.83	82.5	95
	Youtube	0	75	6					
	Dropbox 20	0	67	0					
	Dropbox 120	0	73	0					
	Dropbox 300	0	71	0					
	Owncloud 20	0	66	0					
	Owncloud 120	0	96	0					
	Owncloud 300	0	87	0					
	Propagation Anomaly	1	33	7					
	Netflix	0	510	0	100	100	100	100	
	Propagation Anomaly new test	1	100	0					
10	Browsing	0	696	1	99.49	95.42	99.09	92.06	90
	Youtube	0	684	2					
	Dropbox 20	0	679	0					
	Dropbox 120	0	684	0					
	Dropbox 300	0	733	0					
	Owncloud 20	0	702	0					
	Owncloud 120	0	685	0					
	Owncloud 300	0	684	0					
	Propagation Anomaly	1	313	27					
	Netflix	0	4539	1	99.98	99.94	99.88	100	
	Propagation Anomaly new test	1	840	0					

Ransomware

Table 5.5 shows the result metrics and number of right/wrong predicted classes for the best two slides of ransomware detection using DT algorithm.

Table 5.5: Table with DecisionTreeClassifier results for Ransomware detection

Window slide t	Application	App true class	right prediction	wrong prediction	Accuracy %	F1 %	Precision %	Recall %	PCA variance %
120	Browsing	0	46	5	95.58	77.31	85.17	74	80
	Youtube	0	60	0					
	Dropbox 20	0	73	1					
	Dropbox 120	0	55	2					
	Dropbox 300	0	48	1					
	Owncloud 20	0	50	1					
	Owncloud 120	0	72	0					
	Owncloud 300	0	56	0					
	Owncloud Ransomware	1	18	10					
	Dropbox Ransomware	1	19	3					
	Netflix	0	293	87	79.30	53.11	37.19	96	
	Dropbox Ransomware new test	1	48	2					
60	Browsing	0	110	5	94.27	70.82	76.03	70	80
	Youtube	0	100	5					
	Dropbox 20	0	119	4					
	Dropbox 120	0	122	2					
	Dropbox 300	0	115	1					
	Owncloud 20	0	112	8					
	Owncloud 120	0	101	1					
	Owncloud 300	0	122	3					
	Owncloud Ransomware	1	45	19					
	Dropbox Ransomware	1	25	11					
	Netflix	0	557	203	75.70	50.51	35.45	94	
	Dropbox Ransomware New test	1	94	6					

Theft

Table 5.5 shows the result metrics and number of right/wrong predicted classes for the best two slides of theft detection using DT algorithm.

Table 5.6: Table with DecisionTreeClassifier results for Theft detection

Window slide t	Application	App true class	right prediction	wrong prediction	Accuracy %	F1 %	Precision %	Recall %	PCA variance %
10	Browsing	0	707	0	99.80	75.23	84	70	80
	Youtube	0	670	2					
	Dropbox 20	0	682	0					
	Dropbox 120	0	675	0					
	Dropbox 300	0	672	0					
	Owncloud 20	0	723	0					
	Owncloud 120	0	711	0					
	Owncloud 300	0	708	0					
	Dropbox theft	1	21	9					
	Netflix	0	4539	0	99.67	59	70	53.3	
	Dropbox Theft new test	1	16	14					
20	Browsing	0	344	0	99.42	49	58.33	45	80
	Youtube	0	319	2					
	Dropbox 20	0	351	2					
	Dropbox 120	0	349	0					
	Dropbox 300	0	340	1					
	Owncloud 20	0	362	1					
	Owncloud 120	0	349	0					
	Owncloud 300	0	361	1					
	Dropbox Theft	1	9	11					
	Netflix	0	2270	0	99.52	53.33	70	45	
	Dropbox Theft New test	1	9	11					

DecisionTreeClassifier algorithm conclusions

Table 5.4 shows that DT algorithm can handle propagation anomaly detection, having high accuracy and some wrong predicted classes. Ransomware detection is not so good, the accuracy of new testings is low and results show a high amount of wrong predicted classes. Theft has good accuracy results but shows some wrong predicted classes when detecting theft anomaly.

We will now be testing the same datasets in Gradient Boosting (GB) algorithm in section 5.1.3. We hope for better results, since GB algorithms work in the principle of the ensemble which combines weak learners to get better results.

5.1.3 GradientBoosting

GB works on the principle of the ensemble, it combines a set of weak learners improving prediction and accuracy. At any time instance t, the model outcomes are weighted based on the outcomes of the previous instant t-1. If the outcome is predicted correctly, it is assigned low weight, if not it is assigned high weight. We can divide GB parameters into three categories, Tree-Specific parameters which affect each individual tree, boosting parameters which affect the boosting operation and Miscellaneous parameters that are parameters for overall functioning. Now we will explain the parameters needed to set up the gradient

boosting. We will be using scikit-learn terminologies but the meaning will be the same for any implementation, the name of the parameter might be different from other implementations.

To set up the tree we have:

- `min_samples_split` : Defines the minimum number of samples for a node to be considered for splitting. It is used to control over-fitting. Higher values prevent a model to learn relationships and can lead to under-fitting.
- `min_samples_leaf` : Defines the minimum samples required in a terminal node (leaf). Like `min_samples_split` , it is used to control over-fitting. Low values are good when we have unbalanced classes.
- `min_weight_fraction_leaf` : Similar to `min_samples_leaf` but instead of the number of leaves it is a fraction of the total number of samples. Only one of both `min_samples_leaf and min_weight_fraction_leaf` should be used.
- `max_depth` : Its the maximum depth of a tree. Used to control over-fitting where higher depth values allow the model to learn relations very specific to a particular sample.
- `max_leaf_nodes` : Its the maximum number of terminal nodes (leaves) in a tree. If this parameter is defined, Gradient Boosting Machine (GBM) will ignore `max_depth.`
- `max_features` , Number of features to consider while searhing for the best split. Normally about 30%-40%.

Now for boosting parameters:

- `learning_rate`: It determines the impact of each tree on the final outcome. GBM starts with an initial estimate which is updated using the output of each tree. `learning_rate` controls the magnitude of this change.
- `n_estimators`: Number of trees to be modeled.
- subsample: The fraction of samples to be selected for each tree.

- `learning_rate`: It determines the impact of each tree on the final outcome. GBM starts with an initial estimate which is updated using the output of each tree. `learning_rate` controls the magnitude of this change.
- `n_estimators`: Number of trees to be modeled.
- subsample: The fraction of samples to be selected for each tree.

Now lets look into the miscellaneous parameters:

- loss: It refers to the loss function to be minimized in each split.
- init: Affects the initialization of the output. Can be used as the initial estimates of another model outcome.
- `random_state`: Random number seed so that the same random numbers are generated everytime.
- verbose: The type of output to be printed after the model fits. The different values are:
  - 0: no output
  - 1: output generated for trees in specific intervals
  - >1: output generated for all trees

Table 5.7: Table with GradientBoost results for Propagation detection

Window slide t	Application	App true class	right prediction	wrong prediction	Accuracy %	F1 %	Precision %	Recall %	PCA variance %
10	Browsing	0	692	0	99.58	96.18	99.10	93.53	95
	Youtube	0	687	0					
	Dropbox 20	0	654	0					
	Dropbox 120	0	714	0					
	Dropbox 300	0	702	0					
	Owncloud 20	0	704	0					
	Owncloud 120	0	678	0					
	Owncloud 300	0	716	0					
	Propagation Anomaly	1	318	22					
	Netflix	0	4540	0	100	100	100	100	
	Propagation Anomaly new test	1	840	0					
20	Browsing	0	346	0	99.22	92.53	99.44	87.10	99
	Youtube	0	364	0					
	Dropbox 20	0	313	0					
	Dropbox 120	0	342	0					
	Dropbox 300	0	357	0					
	Owncloud 20	0	345	0					
	Owncloud 120	0	362	0					
	Owncloud 300	0	350	0					
	Propagation Anomaly	1	148	22					
	Netflix	0	2270	0	100	100	100	100	
	Propagation Anomaly new test	1	420	0					

- `warm_start`: Used to fit additional trees on previous fits of a model.
- presort: Used to select wether to presort the data for faster splitting.

GB tests results are shown below.

Propagation

Table 5.7 shows the result metrics and number of right/wrong predicted classes for the best two slides of propagation detection using GB algorithm.

Ransomware

Table 5.8 shows the result metrics and number of right/wrong predicted classes for the best two slides of ransomware detection using GB algorithm.

Table 5.8: Table with GradientBoost results for Ransomware detection

Window slide t	Application	App true class	right prediction	wrong prediction	Accuracy %	F1 %	Precision %	Recall %	PCA variance %
100	Browsing	0	63	0	95.65	78.37	91.15	70	80
	Youtube	0	62	5					
	Dropbox 20	0	61	0					
	Dropbox 120	0	67	0					
	Dropbox 300	0	74	0					
	Owncloud 20	0	82	0					
	Owncloud 120	0	69	0					
	Owncloud 300	0	66	0					
	Owncloud Ransomware	1	25	15					
	Dropbox Ransomware	1	24	6					
	Netflix	0	382	78	84.61	59.40	42.99	96.67	
	Dropbox Ransomware new test	1	58	2					
120	Browsing	0	53	1	97.12	83.42	91.83	78	80
	Youtube	0	52	2					
	Dropbox 20	0	59	0					
	Dropbox 120	0	64	0					
	Dropbox 300	0	60	0					
	Owncloud 20	0	66	0					
	Owncloud 120	0	54	0					
	Owncloud 300	0	58	0					
	Owncloud Ransomware	1	23	9					
	Dropbox Ransomware	1	16	2					
	Netflix	0	298	82	80.23	54.72	39.62	94	
	Dropbox Ransomware New test	1	47	3					

Ransomware

Table 5.9 shows the result metrics and number of right/wrong predicted classes for the best two slides of theft detection using GB algorithm.

Theft

Table 5.9: Table with GradientBoost results for Theft detection

Window slide t	Application	App true class	right prediction	wrong prediction	Accuracy %	F1 %	Precision %	Recall %	PCA variance %
10	Browsing	0	697	0	99.67	63.76	83.33	56.67	80
	Youtube	0	690	4					
	Dropbox 20	0	688	1					
	Dropbox 120	0	729	1					
	Dropbox 300	0	681	0					
	Owncloud 20	0	703	0					
	Owncloud 120	0	674	0					
	Owncloud 300	0	682	0					
	Dropbox theft	1	17	13					
	Netflix	0	4549	0	99.72	57.99	60	56.67	
	Dropbox Theft new test	1	17	13					
20	Browsing	0	373	1	99.32	40.3	35	50	90
	Youtube	0	356	5					
	Dropbox 20	0	347	1					
	Dropbox 120	0	367	0					
	Dropbox 300	0	337	1					
	Owncloud 20	0	338	1					
	Owncloud 120	0	309	0					
	Owncloud 300	0	344	0					
	Dropbox Theft	1	10	10					
	Netflix	0	2033	237	89.38	38.97	34.90	70	
	Dropbox Theft New test	1	14	6					

GradientBoosting algorithm conclusions

Propagation detection in GB algorithm shows good accuracy results and low wrong predicted classes. Ransomware detection shows good results but normal behavior like Netflix shows some wrong predicted classes. Theft detection shows a slide where the results are really good with little wrong predicted classes.

CHAPTER 6

Conclusions and Future work

6.1 Conclusions

The use of machine learning algorithms for network intrusion detection has been studied for the past few years because of its properties, like learning without being explicitly programmed. In chapter 3 we showed some of the approaches that were taken for network anomaly detection. There is no approach that can detect anomalies with 100% accuracy. Also, the most common problem is the number of false positives. Most of the studies that we referenced show as future work the reduction of the number of false positives.

Since we knew what we wanted to detect and we had the means to duplicate the anomalies and normal behavior, we choose the classification approach. We tested some classification algorithms and had some good results in overall.

The detection of propagation anomaly shows better results for the GB algorithm, where we achieve accuracy rates of 99% and better, and low wrong prediction class rates. As for ransomware, SVC is the algorithm that shows better overall results as it detects all Netflix observations and ransomware anomaly with some wrongly predicted classes. GB has better results when it comes to the detection of ransomware anomaly but performs poorly when detecting new normal behavior like Netflix. For theft detection, SVC and DT are the best algorithms, having accuracy rates of 99% and better but have wrong class prediction rates above of what it is expected. Like ransomware, theft anomaly itself is better detected in GB algorithm as it has low false class rates.

6.2 Future Work

For future work, we aim to optimize normal behavior to make it as human like as possible in order to detect not only mechanical made bots but intelligent bots that have a behavior similar to a human. We also aim to increase the number of scenarios and detect more file sharing anomalies. Besides, we want to study a better way of detecting theft attacks as the current approach is not be the best one. Tweak GB algorithm parameters in order to find the

best suitable parameters that have the best performance for our problem is also another goal for future work. Finally, we would also like to study the best way to isolate a machine when an anomaly behavior is detected, as well as study if an outlier machine learning approach performs better than classification.

CHAPTER 7

Attachment

In chapter 5 we presented the best two slide windows or each anomaly and machine learning algorithm. Now in this chapter we present the results of all slide windows and not only the two best. Results are shown in the next sections.

7.1 Propagation

7.1.1 SVM

Table 7.1: Table with all SVM results for propagation detection

Window slide t	accuracy %	F1 %	precision %	recall %	PCA Variance %
10	98.73	87.40	87.40	87.40	95
20	98.37	82.95	100	71.76	
30	97.26	70.50	100	55	
40	97.70	76.10	100	62.22	
50	97.06	64.98	100	50	
60	97.78	75.61	100	63.33	
70	96.59	57.73	100	42	
80	95.73	50.23	90	36	
90	96.57	57.14	100	42.5	
100	96.83	96.83	96.83	96.83	
110	96.36	63.14	100	50	
120	95.80	43	80	30	

7.1.2 GradientBoosting

Table 7.2: Table with all Gradient Boosting results for propagation detection

Window slide t	accuracy %	F1 %	precision %	recall %	PCA Variance %
10	99.58	96.18	99.10	93.53	95
20	99.18	92.40	98.12	87.65	
30	98.68	87.90	96.86	81.67	
40	99.40	94.56	100	90	
50	99.24	92.84	96.25	91.43	
60	99.89	99.66	99.33	100	
70	98.35	84.06	90.17	82	
80	98.13	84.72	91.73	83.99	
90	98.36	83.31	94.17	80	
100	97.50	76.98	94.67	67.5	
110	98.36	86.51	94.67	82.5	
120	98.80	90.64	90.99	93.33	

7.1.3 Tree Classifier

Table 7.3: Table with all Tree Classifier results for propagation detection

Window slide t	accuracy %	F1 %	precision %	recall %	PCA Variance %
10	99.50	95.50	97.12	94.12	95
20	98.98	90.52	96.32	85.88	
30	98.83	89.74	95.18	85.83	
40	98.78	89.89	93.81	87.78	
50	98.40	87.34	89.64	87.14	
60	97.47	80.94	81.17	83.33	
70	98.12	79.90	80.56	82	
80	97.07	76.11	83.81	72	
90	98.06	82.80	86.83	82.5	
100	97.67	77.94	93	70	
110	98.72	89.46	94.66	90	
120	97	76.17	81	76.67	

7.2 Ransomware

7.2.1 SVM

Table 7.4: Table with all SVM results for ransomware detection

Window slide t	accuracy %	F1 %	precision %	recall %	PCA Variance %
10	98.29	89.97	98.41	82.98	95
20	98.50	91.35	98.54	85.52	
30	97.46	85.40	97.67	76	
40	96.43	76.74	100	63.33	
50	96.37	75.98	100	62.5	
60	96.02	73.55	100	59	
70	95.73	71.42	100	57.78	
80	95.64	72.20	100	57.50	
90	94.06	55.77	90	41.43	
100	94.52	66.93	100	51.43	
110	93.33	52.05	100	36.67	
120	94.81	61.90	100	46	

7.2.2 Gradient Boosting

Table 7.5: Table with all Gradient Boosting results for ransomware detection

Window slide t	accuracy %	F1 %	precision %	recall %	PCA Variance %
10	98.91	93.88	97.82	90.35	95
20	98.20	90.09	94.37	86.55	
30	97.95	88.50	96.41	82.5	
40	97.60	86.60	93.51	81.33	
50	96.69	80.81	92.98	71.67	
60	96.02	76.36	90.92	67	
70	96.74	83.06	88.76	78.89	
80	96.54	82.42	87.46	78.75	
90	96.38	79.45	90.92	72.86	
100	96.61	82.38	95	74.29	
110	96.49	80.81	91.83	73.33	
120	97.11	84.49	91.33	80	

7.2.3 Tree Classifier

Table 7.6: Table with all Tree Classifier results for ransomware detection

Window slide t	accuracy %	F1 %	precision %	recall %	PCA Variance %
10	98.99	94.42	96	92.98	95
20	97.98	89.41	88.82	90.34	
30	97.17	85.30	87.73	83.5	
40	96.10	80.12	79.23	82	
50	95.97	79.44	79.98	82.5	
60	96.21	80.31	81.47	80	
70	95.51	78.56	78.41	81.11	
80	95.90	80.47	81.54	81.25	
90	93.77	66.26	69.59	67.14	
100	93.55	69.40	76.30	65.71	
110	96.67	83.35	86.55	83.33	
120	94.61	71.64	76.92	70	

References

[1] M. H. Bhuyan, D. K. Bhattacharyya, and J. K. Kalita, *Network Traffic Anomaly Detection and Prevention*, A. J. Sammes, Ed., ser. Computer Communications and Networks. Cham: Springer International Publishing, 2017, ISBN: 978-3-319-65186-6. DOI: 10.1007/978-3-319-65188-0. [Online]. Available: http://link.springer.com/10.1007/978-3-319-65188-0.

[2] T. M. Mitchell, *Machine Learning*. McGraw-Hill Science/Engineering/Math, 1997, ISBN: 0070428077.

[3] G. Shobha and S. Rangaswamy, "Machine Learning", in, 2018, pp. 197–228. DOI: 10.1016/bs.host.2018.07.004. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/S0169716118300191.

[4] A. Géron, *Hands-On Machine Learning with Scikit-Learn & TensorFlow*, First Edit, N. Tache, Ed. O'Reilly Media, 2017, ISBN: 978-1-491-96229-9.

[5] A. M. R. Heady G. Luger and M. Servillha, "The Architecture of a Network Level Intrusion System", Computer Science Department, University of the New Mexico, Tech. Rep., 1990.

[6] A. Wankhade and K. Chandrasekaran, "Distributed-Intrusion Detection System Using Combination of Ant Colony Optimization (ACO) and Support Vector Machine (SVM)", in *2016 International Conference on Micro-Electronics and Telecommunication Engineering (ICMETE)*, IEEE, Sep. 2016, pp. 646–651, ISBN: 978-1-5090-3411-6. DOI: 10.1109/ICMETE.2016.94. [Online]. Available: http://ieeexplore.ieee.org/document/7938995/.

[7] S. M. K. V. Tan P. N., *Introduction to Data Mining*, P. Education, Ed. Addison-Wesley, 2009.

[8] P. Gogoi, D. K. Bhattacharyya, B. Borah, and J. K. Kalita, "A Survey of Outlier Detection Methods in Network Anomaly Identification", *The Computer Journal*, vol. 54, no. 4, pp. 570–588, Mar. 2011. DOI: 10.1093/comjnl/bxr026.

[9] B. A. Chandola V. and V. Kumar, "Anomaly detection: A survey", *ACM Computing Surveys*, 2009.

[10] L. H. Dash M., "Feature selection for classification", *Intell. Data Anal*, vol. 1, pp. 131–156, 1997.

[11] Y. Chen, Y. Li, X.-Q. Cheng, and L. Guo, "Survey and Taxonomy of Feature Selection Algorithms in Intrusion Detection System", in *Information Security and Cryptology*, Springer Berlin Heidelberg, 2006, pp. 153–167. DOI: 10.1007/11937807_13.

[12] P. L. S. Sarita Tripathy, "A Survey of different methods of clustering for anomaly detection", *International Journal of Scientific & Engineering Reseach*, vol. 6, 2015, ISSN: 2229-5518.

[13] T. Abbes, A. Bouhoula, and M. Rusinowitch, "Efficient decision tree for protocol analysis in intrusion detection", *International Journal of Security and Networks*, vol. 5, no. 4, p. 220, 2010. DOI: 10.1504/ijsn.2010.037661.

[14] C. Wagner, J. François, R. State, and T. Engel, "Machine Learning Approach for {IP}-Flow Record Anomaly Detection", in *{NETWORKING} 2011*, Springer Berlin Heidelberg, 2011, pp. 28–39. DOI: 10.1007/978-3-642-20757-0_3.

[15] B. Schölkopf, J. C. Platt, J. Shawe-Taylor, A. J. Smola, and R. C. Williamson, "Estimating the Support of a High-Dimensional Distribution", *Neural Computation*, vol. 13, no. 7, pp. 1443–1471, Jul. 2001. DOI: 10.1162/089976601750264965.

[16] Q. Yang, H. Fu, and T. Zhu, "An Optimization Method for Parameters of {SVM} in Network Intrusion Detection System", in *2016 International Conference on Distributed Computing in Sensor Systems ({DCOSS})*, IEEE, 2016. DOI: 10.1109/dcoss.2016.48.

[17] H. Kim, S. Kim, M. A. Kouritzin, and W. Sun, "Detecting network portscans through anomoly detection", in *Signal Processing, Sensor Fusion, and Target Recognition {XIII}*, I. Kadar, Ed., SPIE, 2004. DOI: 10.1117/12.546127.

[18] S. Thaseen and C. A. Kumar, "An analysis of supervised tree based classifiers for intrusion detection system", in *2013 International Conference on Pattern Recognition, Informatics and Mobile Engineering*, IEEE, 2013. DOI: 10.1109/icprime.2013.6496489.

[19] J. R. Quinlan, "Improved use of continuous attributes in C4.5", *journal of Artifical Intelligence*, vol. 4, pp. 77–90, 1996.

[20] F. J. Anscombe, "Rejection of Outliers", *Technometrics*, vol. 2, no. 2, pp. 123–146, 1960. DOI: 10.1080/00401706.1960.10489888.

[21] M. Markou and S. Singh, "Novelty detection: a review{\textemdash}part 1: statistical approaches", *Signal Processing*, vol. 83, no. 12, pp. 2481–2497, 2003. DOI: 10.1016/j.sigpro.2003.07.018.

[22] J. P. J. C. J. E. Desforges M. J., "Applications of probability density estimation to the detection of abnormal conditions in engineering", *Proceedings of Institute of Mechanical Engineers*, vol. 212, pp. 687–703, 1998.

[23] E. Eskin, "Anomaly detection over noisy data using learned probability distributions", *Morgan Laufmann Publishers Inc*, 2000.

[24] P. S. Manikopulos C., "Network intrusion and fault detection: a statistical anomaly approach", *IEEE Commun. Mag.*, 2002.

[25] W. Lu and H. Tong, "Detecting Network Anomalies Using {CUSUM} and {EM} Clustering", in *Advances in Computation and Intelligence*, Springer Berlin Heidelberg, 2009, pp. 297–308. DOI: 10.1007/978-3-642-04843-2_32.

[26] Y. Li, X. Luo, Y. Qian, and X. Zhao, "Network-Wide Traffic Anomaly Detection and Localization Based on Robust Multivariate Probabilistic Calibration Model", *Mathematical Problems in Engineering*, vol. 2015, pp. 1–26, 2015. DOI: 10.1155/2015/923792.

[27] K. Wang and S. J.Stolfo, *Anomalous Payload-Based Network Intrusion Detection*. Springer-verlag, 2004.

[28] N. Subramanian, P. S. Pawar, M. Bhatnagar, N. S. Khedekar, S. Guntupalli, N. Satyanarayana, V. K. Vijaykumar, P. K. Ampatt, R. Ranjan, and P. J. Pandit, "Development of a Comprehensive Intrusion Detection System {\textendash} Challenges and Approaches", in *Information Systems Security*, Springer Berlin Heidelberg, 2005, pp. 332–335. DOI: 10.1007/11593980_27.

[29] S. Song, L. Ling, and C. N. Manikopoulo, "Flow-based Statistical Aggregation Schemes for Network Anomaly Detection", in *2006 {IEEE} International Conference on Networking, Sensing and Control*, IEEE, 2006. DOI: 10.1109/icnsc.2006.1673246.

[30] Y. Meng and L.-f. Kwok, "Adaptive context-aware packet filter scheme using statistic-based blacklist generation in network intrusion detection", in *2011 7th International Conference on Information Assurance and Security ({IAS})*, IEEE, 2011. DOI: 10.1109/isias.2011.6122798.

[31] H. Xiao, F. Hong, Z. Zhang, and J. Liao, "Intrusion Detection Using Ensemble of {SVM} Classifiers", in *Fourth International Conference on Fuzzy Systems and Knowledge Discovery ({FSKD} 2007)*, IEEE, 2007. DOI: 10.1109/fskd.2007.371.

[32] V. Timoenko and S. Gajin, "Ensemble classifiers for supervised anomaly based network intrusion detection", 2017.

[33] S. Garg, A. Singh, S. Batra, N. Kumar, and M. Obaidat, "EnClass: Ensemble-based Classification Model for Network Anomaly Detection in Massive Datasets", 2017.

[34] K. Sequeira and M. Zaki, "ADMIT", in *Proceedings of the eighth {ACM} {SIGKDD} international conference on Knowledge discovery and data mining - {KDD}*, {ACM} Press, 2002. DOI: 10.1145/775047.775103.

[35] X.-Q. W. YU-FANG ZHANG ZHONG-YANG XIONG, *Proceedings of the 2005 International Conference on Machine Learning and Cybernetics: August 18-21, 2005, Guangzhou, China, V.1-9.* Institute of Electrical & Electronics Enginee, 2005, ISBN: 0-7803-9091-1. [Online]. Available: https://www.amazon.com/Proceedings-International-Conference-Learning-Cybernetics/dp/0780390911?SubscriptionId=0JYN1NVW651KCA56C102%7B%5C&%7Dtag=techkie-20%7B%5C&%7DlinkCode=xm2%7B%5C&%7Dcamp=2025%7B%5C&%7Dcreative=165953%7B%5C&%7DcreativeASIN=0780390911.

[36] L. Tian and W. Jianwen, "Research on Network Intrusion Detection System Based on Improved K-means Clustering Algorithm", in *2009 International Forum on Computer Science-Technology and Applications*, IEEE, 2009. DOI: 10.1109/ifcsta.2009.25.

[37] M. E. Otey, A. Ghoting, and S. Parthasarathy, "Fast Distributed Outlier Detection in Mixed-Attribute Data Sets", *Data Mining and Knowledge Discovery*, vol. 12, no. 2-3, pp. 203–228, 2006. DOI: 10.1007/s10618-005-0014-6.

[38] M. Ahmed and A. Naser, "A novel approach for outlier detection and clustering improvement", in *2013 {IEEE} 8th Conference on Industrial Electronics and Applications ({ICIEA})*, IEEE, Jun. 2013. DOI: 10.1109/iciea.2013.6566435.

[39] M. H. Bhuyan, D. K. Bhattacharyya, and J. K. Kalita, "A multi-step outlier-based anomaly detection approach to network-wide traffic", *Information Sciences*, vol. 348, pp. 243–271, Jun. 2016. DOI: 10.1016/j.ins.2016.02.023.

[40] N. Duffield, P. Haffner, B. Krishnamurthy, and H. Ringberg, "Rule-Based Anomaly Detection on {IP} Flows", in *{IEEE} {INFOCOM} 2009 - The 28th Conference on Computer Communications*, IEEE, 2009. DOI: 10.1109/infcom.2009.5061947.

[41] G. Liu, Z. Yi, and S. Yang, "A hierarchical intrusion detection model based on the {PCA} neural networks", *Neurocomputing*, vol. 70, no. 7-9, pp. 1561–1568, Mar. 2007. DOI: 10.1016/j.neucom.2006.10.146.

[42] A. J. H. Karimi, "A new Approach for Detecting Intrusions Based on the PCA Neural Networks", *Journal of Basic and Applied Scientific Research*, 2012.

[43] C. Yan, "Intelligent Intrusion Detection Based on Soft Computing", in *2015 Seventh International Conference on Measuring Technology and Mechatronics Automation*, IEEE, Jun. 2015. DOI: 10.1109/icmtma.2015.145.

[44] *HIDE: a hierarchical network intrusion detection system using statistical preprocessing and neural network classification*, IEEE, United State Military Academy, West Point, NY: Workshop on Information Assurance and Security, 2001, pp. 85–90, ISBN: 0780398149.

[45] A. Martin, S. Raponi, T. Combe, and R. Di Pietro, "Docker ecosystem – Vulnerability Analysis", *Computer Communications*, vol. 122, pp. 30–43, Jun. 2018, ISSN: 01403664. DOI: 10.1016/j.comcom.2018.03.011. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/S0140366417300956.