Universidade de Aveiro
2021

Daniel
Martins Coelho

Detecção de Ataques por Canais Laterais na Camada Física

Detection of Side Channel Attacks at the Network Physical Layer

Universidade de Aveiro
2021

Daniel Martins Coelho

Detecção de Ataques por Canais Laterais na Camada Física

Detection of Side Channel Attacks at the Network Physical Layer

"*Working hard for something we don't care about is called stress; working hard for something we love is called passion.*"

— Simon Sinek

Universidade de Aveiro
2021

Daniel
Martins Coelho

Detecção de Ataques por Canais Laterais na Camada Física

Detection of Side Channel Attacks at the Network Physical Layer

Dissertação 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 Paulo Jorge Salvador Serra Ferreira, Professor Associado do Departamento de Eletrónica, Telecomunicações e Informática da Universidade de Aveiro.

Dedico este trabalho aos meus avós Ermelinda e Arlindo, aos meus pais e aos meus amigos por todo o apoio que me deram nestes cinco anos. A preocupação e o carinho foram fundamentais para passar por todo o percurso com distinção.

agradecimentos / acknowledgements

Um forte agradecimento por todo o apoio que a minha família me deu durante todo o meu percurso académico, em especial aos meus pais e avós, que fizeram os possíveis para me manter focado e saudável. Sempre foram uma fonte de inspiração e suporte incrível. Quero também agradecer aos meus amigos, pela amizade e o carinho. Acompanham-me à vários anos e estiveram cá para mais uma etapa na minha vida.
Gostava também de agradecer à Scientific Junior Value, uma júnior empresa na qual cresci como pessoa e profissional e que para sempre guardo todos os amigos que fiz pertencendo a esta júnior empresa.
Quero também deixar um especial agradecimento ao Professor Paulo Salvador, que desde a primeira aula, no inicio do curso, sempre mostrou disponibilidade de ajudar, sempre transmitiu ideais de como ser um bom profissional e mostrou ser uma ótima pessoa. Para além disso, foi incansável o seu apoio para a realização desta dissertação.
Por isso, do fundo do coração um grande obrigado a todos.

Palavras Chave

Canais Laterais, Monitorização de Rede, Monitorização do Sinal Rádio, Deteção de Anomalias, Classificação Mono-Classe, Aprendizagem Automática.

Resumo

Hoje, com o advento da IoT e a resultante fragmentação das tecnologias sem fio, elas trazem não apenas benefícios, mas também preocupações. Diariamente vários indivíduos se comunicam entre si usando vários métodos de comunicação. Os indivíduos usam uma variedade de dispositivos para atividades inócuas do dia-a-dia; no entanto, existem alguns indivíduos mal-intencionados (agentes desonestos) cujo objetivo é causar danos, sendo a exfiltração de informação uma das maiores preocupações. Sendo a segurança das comunicações Wi-Fi uma das áreas de maior investimento e investigação no que toca a segurança na Internet, os agentes desonestos fazem uso de canais laterais para exfiltrar informação, nomeadamente o Bluetooth. A maioria das soluções atuais para deteção de anomalias em redes baseiam-se em analisar tramas ou pacotes, o que, inadvertidamente, pode revelar padrões de comportamento dos utilizadores, que estes considerem privados. Além disso, as soluções que se focam em inspecionar dados da camada física normalmente usam a potência de sinal recebido (RSSI) como uma métrica de distância e detetam anomalias baseadas na posição relativa dos nós da rede, ou usam os valores do espetro diretamente em modelos de classificação sem prévio tratamento de dados.

Esta Dissertação propõe mecanismos para deteção de anomalias, assegurando simultaneamente a privacidade dos seus nós, que se baseiam na análise de atividade rádio na camada física, medindo os comportamentos da rede através do número de frequências ativas e inativas e a duração de períodos de silêncio e atividade. Depois da extração de propriedades que caracterizam estas métricas, é realizada uma exploração dos dados e um estudo das mesmas, sendo depois usadas para treinar modelos de classificação mono-classe.

Os modelos são treinados com dados retirados de uma série de interações entre um computador, um AP, e um telemóvel num ambiente com ruído reduzido, numa tentativa de simular um cenário de automação doméstica simplificado. De seguida, os modelos foram testados com dados semelhantes mas contendo um nó comprometido, que periodicamente enviava um ficheiro para uma máquina local através de uma ligação Bluetooth. Os dados mostram que, em ambas as situações, foi possível atingir taxas de precisão de deteção na ordem dos 75% e 99%.

Este trabalho finaliza com algumas ideias de trabalho futuro, nomeadamente alterações ao nível do pré-processamento, ideias de novos testes e como diminuir a percentagem de falsos negativos.

Keywords

Side-Channels, Network Monitoring, Radio Signal Monitoring, Anomaly Detection, One-Class Classification, Machine Learning.

Abstract

Today, with the advent of IoT and the resulting fragmentation of wireless technologies, they bring not only benefits, but also concerns. Daily, several individuals communicate with each other using various communication methods. Individuals use a variety of devices for innocuous day-to-day activities; however, there are some malicious individuals (dishonest agents) whose aim is to cause harm, with the exfiltration of information being one of the biggest concerns. Since the security of Wi-Fi communications is one of the areas of greatest investment and research regarding Internet security, dishonest agents make use of side channels to exfiltrate information, namely Bluetooth. Most current solutions for anomaly detection on networks are based on analyzing frames or packets, which, inadvertently, can reveal user behavior patterns, which they consider to be private. In addition, solutions that focus on inspecting physical layer data typically use received signal power (RSSI) as a distance metric and detect anomalies based on the relative position of the network nodes, or use the spectrum values directly on models classification without prior data processing.

This Dissertation proposes mechanisms to detect anomalies, while ensuring the privacy of its nodes, which are based on the analysis of radio activity in the physical layer, measuring the behavior of the network through the number of active and inactive frequencies and the duration of periods of silence and activity. After the extraction of properties that characterize these metrics,an exploration and study of the data is carried out, followed by the use of the result to train One-Class Classification models.

The models are trained with data taken from a series of interactions between a computer, an AP, and a mobile phone in an environment with reduced noise, in an attempt to simulate a simplified home automation scenario. Then, the models were tested with similar data but containing a compromised node, which periodically sent a file to a local machine via a Bluetooth connection. The data show that, in both situations, it was possible to achieve detection accuracy rates in the order of 75 % and 99 %.

This work ends with some ideas of resource work, namely changes in the level of pre-processing, ideas of new tests and how to reduce the percentage of false negatives.

Contents

List of Figures

List of Tables

Glossary

IoT	Internet of Things
MAPI	Manufacturers Alliance for Productivity and Innovation
OWASP	Open Web Application Security Project
MRMR	Minimum Redundancy-Maximum Relevance
PCA	Principal Component Analysis
OCC	One-Class Classification
EM	Expectation-Maximization
ETC	Extra Tree Classifier
CV	Cross-validation
OC-SVM	One-Class Support Vector Machine
IF	Isolation Forest
LOF	Local Outlier Factor
KDE	Kernel Density Estimation
GMM	Gaussian Mixture Model
RF	Radio Frequency
UTM	Unified Threat Management
IPS	Intrusion Prevention System
IDS	Intrusion Detection System
NIDS	Network Intrusion Detection System
HIDS	Host-based Intrusion Detection System
BLE	Bluetooth Low Energy
APT	Advanced Persistent Threat
CVE	Common Vulnerabilities and Exposures
C&C	Command and Control
P2P	Peer-to-Peer
DDoS	Distributed Denial-of-Service
OWASP	Open Web Application Security Project
SIG	Special Interest Group
ISM	Industrial, Scientific and Medical
EDP	Energias de Portugal
SDR	Software Defined Radio
FFT	Fast Fourier Transform
LNA	Low-Noise Amplifier
VGA	Variable-Gain Amplifier
AP	Access Point
EMD	Empirical Mode Decomposition
IMF	Intrinsic Mode Function
QoS	Quality Of Service
SNMP	Simple Network Management Protocol
SSH	Secure Socket Shell
DNS	Domain Name Service
kNN	k-Nearest Neighbours
ML	Machine Learning
NIDS	Network-based Intrusion Detection System
OCC	One-Class Classification
RSSI	Received Signal Strength Indicator
SVM	Support Vector Machine

CHAPTER 1

Introduction

This chapter describes the overall motivation for addressing the topics of this Dissertation, the background of the Internet of Things (IoT), parallel data exfiltration and machine learning areas. In this chapter, we also address the main contributions of this work and a summary of the structure of the document.

1.1 Motivation

IoT devices have become an important part of the information and communication technology performing a crucial role in a company make them more efficient and productive, especially if they assist curtail on manual steps and human error. However, the growing complexity in IoT infrastructures is raising unwanted vulnerability to their systems. IoT devices bring indisputable benefits to the companies - which are undeniably numerous - but they often forget to simultaneously assess the risks of information breaches related to those devices. According to ESET's [1] 2020 threat report says that all top ten vulnerabilities in Q2 2020 originated from before 2016, demonstrating the "longevity" of IoT flaws and the reluctance or inability of vendors and/or users to patch them [1]. In IoT devices, security breach and anomaly has become common phenomena nowadays. IoT devices use a wireless medium to broadcast data which makes them an easier target for an attack [2]. A normal communication attack in the local network is limited to local nodes, but attack in IoT ecosystem expands over a larger area and has a devastating effects. The security measures that have been used until now are weak against the vulnerability of IoT devices. For companies, data is money. Vulnerability in IoT nodes makes a backdoor for an attacker to data exfiltration from any important organization [3]. According to IBM, the average total cost per data breach worldwide in 2019 amounted to a total of $3.92 million and $3.5 million in 2014 [4].

[1]https://www.eset.com/

Conventional wireless security solutions specialize in perimeter network defense, like Unified Threat Management (UTM), Intrusion Prevention System (IPS), and authentication solutions, being great at preventing, detecting, and monitoring threats coming over the network, however they are inept at protecting against IoT-related attacks. Unlike ancient IT security exploits, IoT threats gain enterprise access through the broader Radio Frequency (RF) spectrum. It is not just a laptop or smartphone accessing corporation WiFi that presents a threat; it's any device enabled by Bluetooth, NFC, RFID, among others. It is no longer sufficient to protect only the perimeter of the company's network. Companies need to be aware that the coming of IoT devices brings a greater number of communication channels. If one were breached, it might be devastating to the corporation.

Nowadays, the companies are a strong interest in detecting an attack after compromising the network instead of preventing it from occurring. According to a global study of large organizations in 2019, presented by Visual Capitalist[2], there are 6.5 million data records compromised every day, 42% are caused by criminals. In 2014, the Syrian Electronic Army hacked eBay and had full access to its servers for 229 days. The hacking compromised 145 million accounts. [5]. For companies, a data breach can mean the release publicly of sensitive information, proprietary trade secrets, and customer data, resulting in hefty fines and damage to the company. Many companies are therefore interest in detecting and preventing breaches as they happen. To do that we need to identify the unauthorized transfer of data from inside, known as data exfiltration.

A network typically has a large volume of data in circulation, which the vast majority is legitimate. Identifying malicious data, while avoiding false positives, is the challenge for building an exfiltration detector. Machine learning is a strong and useful tool for these scenarios. Anomaly detection can automate the search for outliers, and separate them out from the traffic that warrants further investigation. There are several ways to increase the robustness of the resulting alerts. It's used statistical and temporal characteristics of the RF spectrum to define a typical behavior of the network and facilitate anomaly detection.

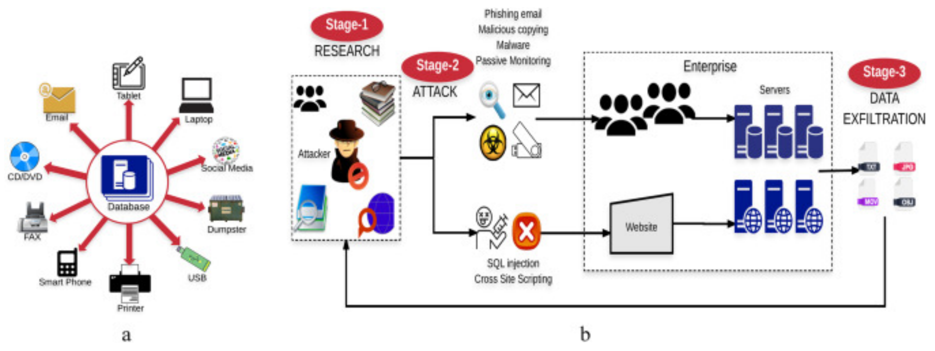


Figure 1.1: Stages of an attack with exfiltration data (retrieved from article [6]).

[2]https://www.visualcapitalist.com/

1.2 Objectives

The most popular network monitoring systems and tools focus primarily on measurements and tracking of network status, there is an enormous security gap in the physical layer [7]. Furthermore, modeling physical layer behavior is not as easy as on the upper layers. First, one can not discriminate individual flows of data like when inspecting the network layers. We are not provided with as much data as on upper layers. In the physical layer, the only available data are power indicators throughout the time, which makes it much more challenging to differentiate network behavior.

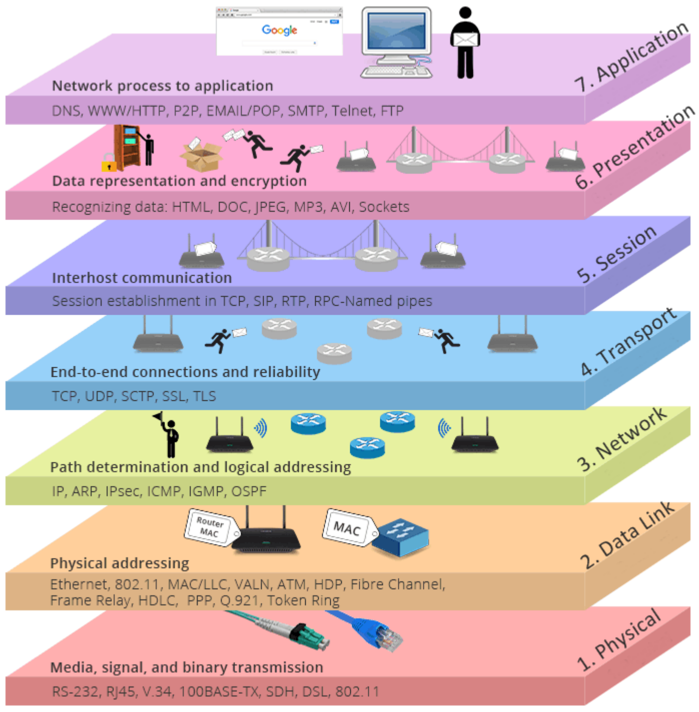

7. Application
Network process to application
DNS, WWW/HTTP, P2P, EMAIL/POP, SMTP, Telnet, FTP
6. Presentation
Data representation and encryption
Recognizing data: HTML, DOC, JPEG, MP3, AVI, Sockets
5. Session
Interhost communication
Session establishment in TCP, SIP, RTP, RPC-Named pipes
4. Transport
End-to-end connections and reliability
TCP, UDP, SCTP, SSL, TLS
3. Network
Path determination and logical addressing
IP, ARP, IPsec, ICMP, IGMP, OSPF
2. Data Link
Physical addressing
Ethernet, 802.11, MAC/LLC, VALN, ATM, HDP, Fibre Channel, Frame Relay, HDLC, PPP, Q.921, Token Ring
Router MAC
MAC
1. Physical
Media, signal, and binary transmission
RS-232, RJ45, V.34, 100BASE-TX, SDH, DSL, 802.11

Figure 1.2: OSI Model (retrieved from website [8]).

This dissertation aims at detecting anomalous behavior in IoT networks without compromising user or node privacy using the physical layer. Even if the majority of current solutions do not study individual users' behavior on the network, they do not guarantee their privacy since they have access to data that may disclosure it.

Traditionally, Network-based Intrusion Detection System (NIDS) are broadly classified based in two different ways: systems counting on misuse-detection monitor activity with precise descriptions of known malicious behavior, while anomaly-detection systems have a perception of regular activity and flag deviations from that profile. Compared to other security areas on research world, anomaly detection has not obtained much traction in the real world. Those systems found in operation deployment are most commonly based on statistical profiles of heavily aggregated traffic [9]. According to Robin Sommer and Vern Paxson, in terms of actual deployments, we observe a striking imbalance: in operational settings, they find almost exclusively only misuse detectors in use—most commonly in the form of signature systems that scan network traffic for characteristic byte sequences [9]. Robin Sommer and Vern Paxson speculate that many anomaly detection systems in the academic world do not live up to the requirements of operational settings.

First introduced by Denning in her seminal work on the host-based IDES system [10] in 1987, to capture normal activity, IDES (and its successor NIDES [11]) used a combination of statistical metrics and profiles. Since then, many more approaches have been proposed. Often, they tested several techniques from the machine learning community, such as information theory [12], neural networks [13], support vector machines [14], genetic algorithms [15], among others. In our discussion, we focus on anomaly detection systems that utilize some of this machine learning approaches.

This dissertation is focused on defining methodologies to extract the important information, processing data and investigating the potential of machine learning techniques to increase the safety of IoT devices, focusing on the development of unsupervised learning techniques. Finally, it discusses the results of the overcome challenges and the paths that can be taken to effectively improve the detection of anomalies so that it can perform a good job of protecting IoT devices.

1.3 Contributions

Taking into account the current security challenges IoT networks face, it was developed work that can be summarized into four points:

- Using a Software-Defined Radio (SDR) it was developed a monitoring system of the network interactions at layer 2.
- Descriptive data from the network were extracted from the monitoring system, namely Wi-Fi and Bluetooth, which were cleaned, filtered and selected.
- Accurately model anomaly-free network behavior, and detect a broad range of outliers, by using classical approaches of One-Class Classification (OCC) models.
- Validate the proposed mechanisms by designing one use-case where the goal was to identify periodic outlier behavior on Bluetooth. The behavior of the tampered node is masked by data exchanges of other devices in the network.

1.4 Document Structure

This document is structured as follows:

- **Chapter 2 - State of the Art**: it presents the background of current issues and solutions to secure IoT networks, as well as current works regarding physical layer monitoring. Furthermore, it addresses machine learning techniques currently used for OCC.
- **Chapter 3 - Scenario and Methodologies for Anomaly Detection**: this chapter showcases the proposed mechanisms for detecting anomaly in physical layer. It depicts the

entire data pipeline, starting with the characterization of the scenario to be studied and finding activity and silence periods and the number of active and inactive frequencies, and extracting relevant resources from those metrics for classification. It also demonstrates how those features should be treated, and how to properly train the OCC models that will be in charge of labeling each sample.

- **Chapter 4 - Methodologies Evaluation**: it describes two scenarios with few devices presenting a home IoT scenario in both, however one with more data volume, which was designed to validate the mechanisms presented in the previous chapter. It showcases a carefull RSSI and feature analysis, and the results that OCC models obtained.
- **Chapter 5 - Conclusions and Future Work**: this chapter summarizes the work presented through chapters two to four, and depicts the final conclusions of this dissertation. It finalizes this dissertation by proposing future developments in the issued area and a scalable framework for real-time network behavior classification, by gathering RSSI data from different network probes and labelling it in a separate server.

CHAPTER 2

State of the Art

This chapter presents the state of the art and the background of the main topics addressed by this work. The first regards the current state of IoT networks, more explicitly detailing its story, and security issues and vulnerabilities, many due to its devices heterogeneous, lack of concern about security mechanisms, and lack of awareness of the imminent dangers.

The second part focuses on the most common wireless communication channels used in IoT, respectively, WiFi and Bluetooth. Its discussed its technical features, how important they are for IoT environments, and why Bluetooth is becoming the most adopted solution.

The third section presents some methodologies used in attacks, focusing on zero-day attacks and exfiltration of data through IoT. Based on the assumption that the network is already compromised by this proof of concept. The phases of an attack are described in order to understand how this happens.

The fourth section, focuses on monitoring, exploring how data is collected for monitoring, deepening the concept of SDR. Also, it is presented methods to detect anomalies in the network, as well as the types of intrusion detection systems and what machine learning approaches have been tested in the research world using machine learning for detecting outlier behaviors in the network.

The final section illustrates a series of machine learning techniques for anomaly detection, particularly OCC models, and feature engineering techniques, such as feature selection and feature dimensionality reduction. Many of these techniques will be used for anomaly detection of physical layer behavior, as described in the following chapters.

2.1 Internet of Things

The Internet of Things, also known by IoT, it comprises all devices and objects that are enabled to be permanently connected to the Internet, being able to identify themselves on the network and communicate with each other. They may have their state altered by the environment around them, with or without the active involvement of human beings and they are able to collect a vast amount of information about those around them. The Internet Society [16] broadly defines IoT as "*scenarios where network connectivity and computing capability extends to objects, sensors and everyday items not normally considered computers, allowing these devices to generate, exchange and consume data with minimal human intervention*" .

Vehicles, traffic lights, surveillance cameras, environment sensors and medical devices are just a few examples of what already exists in the IoT world. The benign objective of all these devices, and the large amount of data resulting from interaction through the Internet, is that the resulting processing is carried out so that, for example, traffic jams are avoided or a disease is anticipated, to give just a few examples.

This new technological wave is much more than the mere implementation of electronic systems in general in the production processes in the factories as characterized by Industry 3.0, it is based on the concept explained above about IoT, enabling a great interaction between different devices along the production chain, including the logistics chain, providing that the manufacturing processes result in communion between the physical and virtual world.

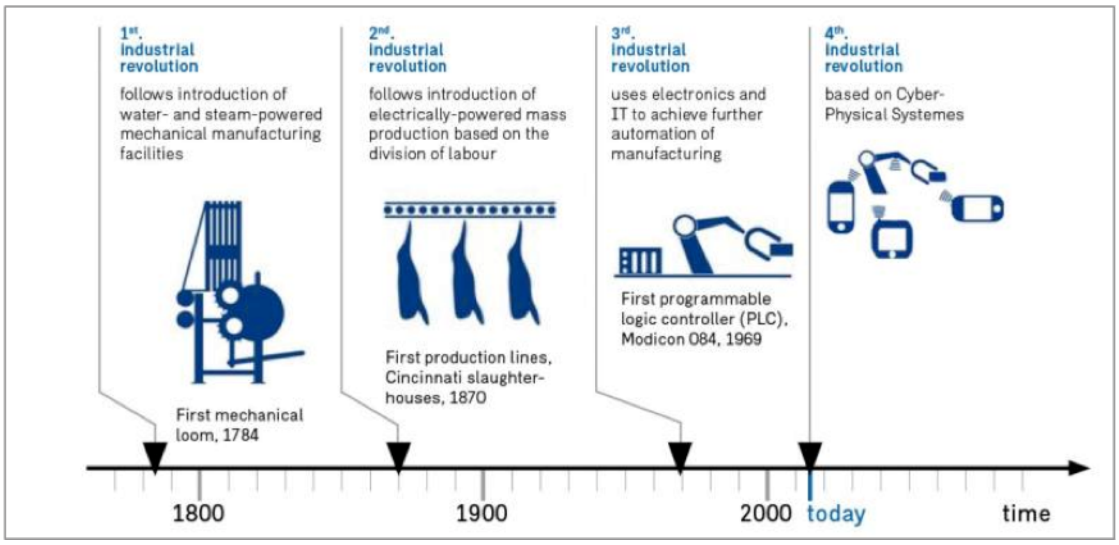

1st. industrial revolution
follows introduction of water- and steam-powered mechanical manufacturing facilities
First mechanical loom, 1784
2nd. industrial revolution
follows introduction of electrically-powered mass production based on the division of labour
First production lines, Cincinnati slaughter-houses, 1870
3rd. industrial revolution
uses electronics and IT to achieve further automation of manufacturing
First programmable logic controller (PLC), Modicon 084, 1969
4th. industrial revolution
based on Cyber-Physical Systemes
1800
1900
2000
today
time

Figure 2.1: Industrial Revolution (retrieved from [17])

In other words, the industrial revolution 4.0, both in the production component and in the logistics component, contemplates the symbiosis of digital information from various sources and locations, in order to command and control the physical act of producing and distributing a product or a set of products.

Thus, IoT is an irreversible reality and will gradually invade the world, our homes, the spaces where we work, and our personal and professional life. Therefore, we can either ignore or prepare better for the new normality. The IoT concept does not only bring benefits but also harm, both for companies and for the personal life of each one. The number of devices connected to the Internet, which interact with the larger world, also increases the risk of suffering a serious compromise due to the inherent and substantial attack surface.

2.1.1 IoT Security Issues

A work developed by *Deloitte* in partnership with the Manufacturers Alliance for Productivity and Innovation (MAPI) demonstrates that almost half of the manufacturers have mobile applications associated with their products, with **76% of companies choosing WiFi** as a data flow transmission channel between their products, easily eclipsing **the use of Bluetooth, 48%**. According to the same article, "*Over half of the manufacturing executives, 52% surveyed said the connected products their companies produce are able to store and/or transmit confidential data, including social security and banking information.*", with only 55% of manufacturers saying they use encryption as the most widely used method to protect data as it flows through their devices. Using this data as a basis, the lack of data security mechanisms creates considerable vulnerabilities for those who have these objects installed in their workspaces and homes. In cases of cyber breaches, 40% of manufacturers do not include these products in their responses to incidents. In short, it is imperative that manufacturers produce this type of product, have "security by design" as a standard and whoever acquires them requires evidence of this. [18].

The growth in remote work is compounded by a worldwide trend: the escalation of the internet of things (IoT) devices at our homes and work. In fact, one trillion IoT devices are expected by 2025 [19], and most security teams have zero visibility into them. As consumer IoT devices increasingly share the same network as corporate devices, consumer IoT devices effectively expand the organization's attack surface and exacerbate this growing blind spot.

According to the newspaper *PÚBLICO*, "*'The pandemic is causing the largest volume of attacks we have seen at one time,' says Mark Rogers to the PÚBLICO, one of the founders of the CTI League* [1]." [20]. A company with several employees working from home, it is certain that they will access several applications simultaneously and the security systems will not be able to filter everything. An "infected" mobile phone can steal credentials and access the network and then have access to the work computer, for example, and thus access the work network [20]. Without a federally enforced standard governing the security of consumer IoT devices, device manufacturers have been permitted to prioritize time-to-market above security, it leaves massive potential for hackers to take over devices and use them for cyber attacks [21].

A 2018 Worlwide Thread Assessment from Robert Ashley, director of the US Defense Intelligence Agency, warned that weak security on IoT devices posed one of the "*most important emerging cyber attacks*" to national security [22].

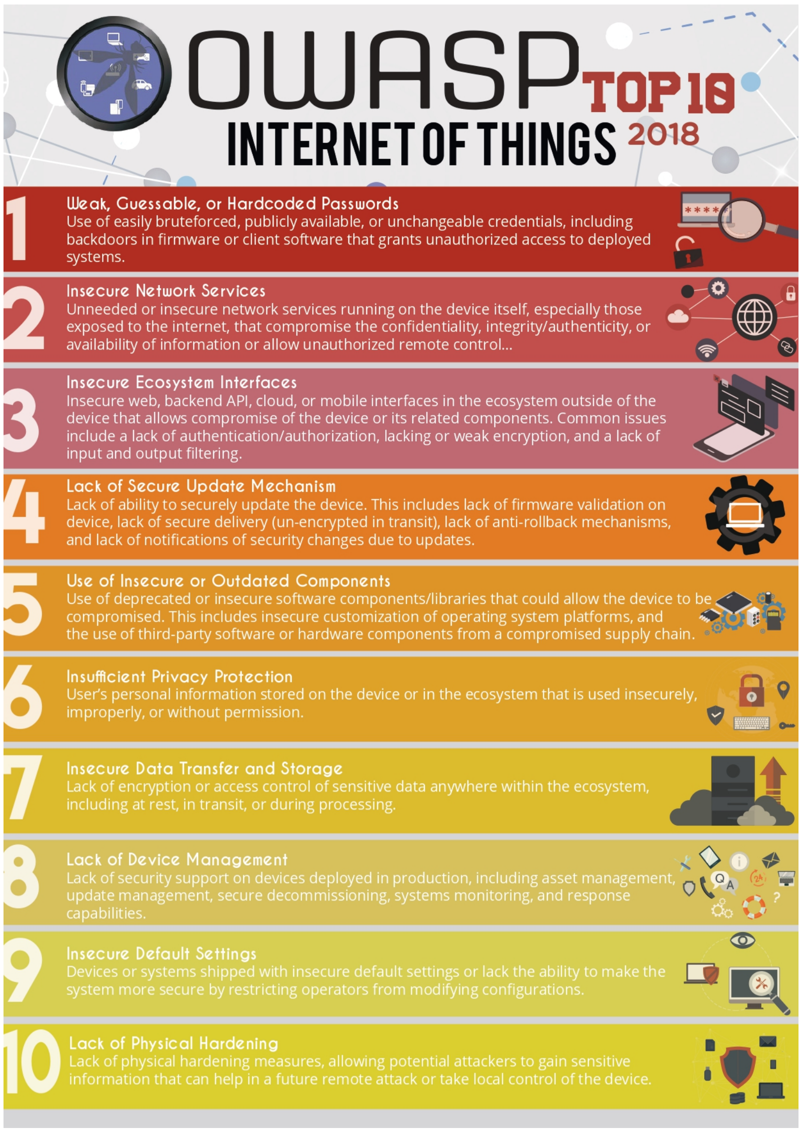
2.1.2 Vulnerabilities in IoT devices

The variety of IoT devices is wide and accessible to anyone, from IP cameras, lights or thermostats. Unlike previous IT technologies, the security analysis of IoT devices is complex due to the wide variety of devices and software available on the market. While "things" on the Internet of Things benefit homes, factories and cities, these devices can also present blind spots and security risks in the form of vulnerabilities.

[1] https://cti-league.com/

Some of the explanations why IoT devices are vulnerable are, **limited computational abilities and hardware limitations**, since the devices have specific functions that warrant only limited computational abilities, leaving little room for robust security mechanisms and data protection, **heterogeneous transmission technology**, turn it difficult to determine standard protection methods and protocols, **components vulnerability**, and **users lacking security awareness**, users awareness could expose smart devices vulnerabilities to threads and attack openings.

The OWASP foundation, in 2018, developed a Top 10 related to the vulnerabilities of IoT devices and that should be taken into account by all those who develop, create or manage IoT systems.



OWASP TOP 10
INTERNET OF THINGS 2018

1 Weak, Guessable, or Hardcoded Passwords
Use of easily bruteforced, publicly available, or unchangeable credentials, including backdoors in firmware or client software that grants unauthorized access to deployed systems.

2 Insecure Network Services
Unneeded or insecure network services running on the device itself, especially those exposed to the internet, that compromise the confidentiality, integrity/authenticity, or availability of information or allow unauthorized remote control...

3 Insecure Ecosystem Interfaces
Insecure web, backend API, cloud, or mobile interfaces in the ecosystem outside of the device that allows compromise of the device or its related components. Common issues include a lack of authentication/authorization, lacking or weak encryption, and a lack of input and output filtering.

4 Lack of Secure Update Mechanism
Lack of ability to securely update the device. This includes lack of firmware validation on device, lack of secure delivery (un-encrypted in transit), lack of anti-rollback mechanisms, and lack of notifications of security changes due to updates.

5 Use of Insecure or Outdated Components
Use of deprecated or insecure software components/libraries that could allow the device to be compromised. This includes insecure customization of operating system platforms, and the use of third-party software or hardware components from a compromised supply chain.

6 Insufficient Privacy Protection
User's personal information stored on the device or in the ecosystem that is used insecurely, improperly, or without permission.

7 Insecure Data Transfer and Storage
Lack of encryption or access control of sensitive data anywhere within the ecosystem, including at rest, in transit, or during processing.

8 Lack of Device Management
Lack of security support on devices deployed in production, including asset management, update management, secure decommissioning, systems monitoring, and response capabilities.

9 Insecure Default Settings
Devices or systems shipped with insecure default settings or lack the ability to make the system more secure by restricting operators from modifying configurations.

10 Lack of Physical Hardening
Lack of physical hardening measures, allowing potential attackers to gain sensitive information that can help in a future remote attack or take local control of the device.

Figure 2.2: Top 10 security issues associated with IoT devices developed by OWASP in 2018 (retrieved from [23])

An example of the use of these vulnerabilities is in the 2015 incident, in which an automotive company sold more than a million vehicles with vulnerabilities that could be exposed by jackware [2], as was the case with the Fiat Chrysler Jeep. [24].

Although most research and discussion on "car hacking" is about technical issues inside the vehicle, it's important to understand that much of IoT technology depends on a support system that goes far beyond the device itself which can cause remote access from devices.

It is possible to confirm this with the news from 2015 with VTech, which operates in the area of IoT for children (IoCT). A security breach in electronic toys exposed personal data about the kids and their parents, showing everyone how many attack possibilities are created through IoT. According to Troy Hunt, a security expert, called by VTech to analyze the problem, concludes that he found almost 5 million accounts with weak and easy to crack passwords, and the secret questions used to recover passwords were kept in a simple format of text [25].

2.2 Wireless Communication Technologies

Different Wireless Communication Technologies can be used for connecting the IoT devices as local networks, and connecting these local networks (or individuals IoT devices) to the Internet.

2.2.1 WiFi

WiFi is the most popular wireless technology used in IoT environments. It's based on IEEE 802.11 standards. It can operate in 2.4GHz, 3.5GHz, and 5GHz unlicensed Industrial, Scientific and Medical (ISM) frequency bands. It provides a coverage range of up to 100 meters. It also provides reliable, secure, and high-speed communications. However, it supports short-range communications. The cost and power of WLAN products are also higher than other short-range wireless technologies such as ZigBee, Z-Wave, or Bluetooth [26].

WiFi plays a task in most IoT environments, alone or interworking with more specialized protocols, or with cellular. Some IoT applications, like vehicular services, or video-based apps like connected security cameras, will need the bandwidth of the wireless broadband network, implemented to enable other requirements like low latency. In critical environments, low latency may be a crucial condition. WiFi is uniquely placed to support broadband and narrowband IoT applications from a standard platform which will work on several levels of power consumption and signal range. Subsequent release of 5G standards, will prioritize IoT-focused capabilities like latency below four milliseconds and really high availability, to support emerging cases within the URLLC (ultra-reliable low latency communications) category [27].

[2]Jackware acts the same as ransomware. When installed on a device, it will lock access and hold the device for ransom. It differs from ransomware in that it can disallow control to an entire device.

2.2.2 Bluetooth

Bluetooth is a technology developed 26 years ago, in 1994, when the first draft of data transmission would begin to develop to become one of the main pillars of IoT. In 1998, the Bluetooth Special Interest Group (SIG) was formed, which until today publish and promotes the standard ans its subsequent revisions [28].

The technology once pronounced dead by a trade magazine journalist in 2003 [29] was been prosperity in the last 10 years for audio communications and stereo streaming. Daniel Kleiner from Bluetooth SIG [3] said that is projected that nearly 4.6 billion Bluetooth devices will ship in 2019 [30].

Bluetooth enables low power communication between devices that are in close proximity of each other. It operates in the unlicensed ISM RF 2.4 GHz spectrum. Bluetooth is used for data transmission through short range UHF radio waves between 2.4 and 2.485GHz, occupying very similar frequencies with WiFi, although has always been designed as a much shorter range and lower power alternative [31].

One main advantage of the technology is its ability to transmit both voice and data simultaneously. Bluetooth devices operate on 79 different frequencies in the radio wave spectrum. When two devices wish to connect, they choose one of the 79 channels at random, or try a different one if it is already occupied by another pair of nearby devices.

2.2.3 Choice of Bluetooth over WiFi in IoT environments

Both Bluetooth and WiFi are commonly used for IoT devices, but there are advantages and drawbacks to both choices because they operate in different ways.

The Bluetooth protocol has two different versions commonly employed by IoT devices which will indirectly communicate with each other: Bluetooth Classic and Bluetooth Low Energy (BLE), which is meant for devices that require to consume low amounts of power. Bluetooth usually requires physical proximity to initiate a sign broadcast, so there's no possibility of remote attacks, also requires much less energy than WiFi, so it works better for low-power IoT devices like basic sensors [32]. Also, the BLE provides a totally new approach in terms of cyber-securing the network. The previous version of Bluetooth was known to contain variety of security vulnerabilities that would cause exposure of encryption keys [33].

Another advantage of Bluetooth networking is that the incontrovertible fact that it's supported by the overwhelming majority of the mobile phones that are in use today. It's therefore easy to use the resident's smartphone to supply an easy interface with the smart devices installed within the premises. Moreover, Bluetooth allows for greater localization accuracy compared to WiFi, thanks to its more limited range. It's also easier and safer to line up and operate, thanks to the inherent features in Bluetooth's design [33].

Properly-designed Bluetooth IoT applications are effective for several specific application domains. Since Bluetooth is pervasive in smartphones and private computers, it's gaining ground in home automation applications albeit the trouble to be a very low-power technology

[3]https://www.bluetooth.com/

remains ongoing [34]. The Bluetooth immediately pulled in financial specialists, who look with revenue to the expanding business sector of the IoT.

2.3 Attack Methodologies

Targeted cyber attacks are a standard weapon for subverting the integrity of Internet operations. Those operations steal intellectual property, conduct espionage, damage critical infrastructures, and make danger for users. It are able to do both tactical and strategic goals across the internet without requiring any physical requirement. It's clear that targeted attacks are a strong weapon playing a big role in future cyber wars [35].

2.3.1 Zero-day Attack

Intruders are smarter and their methods are unpredictable. They will eventually find a way to penetrate the defenses. This means that at some point the preventive measures will fail. The best intruders save their exploits for the targets that truly matter, zero-day exploits.

Zero-day exploits are exploits that are designed for unknown vulnerabilities that vendors haven't any awareness and no patches are available. In critical software is now considered to be attack weapons which will be wont to gain control of an opponent's network infrastructure.

Government security agencies are spending many dollars on unknown zero-day exploits. The United States government is one among the most important buyers of those cyber weapons [36]. In fact, legitimate security companies find vulnerabilities, write zero-day exploits, and sell them to governments for giant amounts of cash. A well-crafted targeted cyber attack isn't cheap as substantial effort is expended in building a multi-layer model of attack vectors and adapting them to the target network's environment. Targeted attacks are nation independent and may be initiated by independent attackers [35].

Patching may be a bigger problem than zero-day exploits. In many cases, like with IoT devices, they're shipped from the factory in a vulnerable state, then never get patched. Sometimes is physically impossible to patch these devices. Although a security patch has been published by the seller does little good if that patch never gets deployed in production [37].

2.3.2 Targeted Attacks

There exist several definitions of targeted cyber attacks, however, the basic definition based on the naming convention says that a targeted attack is a class of dedicated attacks that aim at a specific user, company, or organization to gain access to critical data. Targeted-attacks differentiate the targets and wait for the appropriate opportunity to execute the attack plan, unlike broad-based attacks that are random in nature and focus on infecting and compromising large groups of users [35].

The targeted attacks follow some important characteristics as:

- Zero-day exploits against unknown vulnerabilities are used to compromised target systems.

- Sophisticated malware are used, which go unnoticed despite the presence of security solutions installed on network.
- Real identity behind the attack is hidden to keep a low profile to avoid any legal problems.
- Only the targeted systems are compromised and infected. A targeted attack is less likely to be caught and becomes more dangerous.
- Attack is made persistent for a long period of time and operations are executed in a hidden manner.

Targeted attacks are complex because attackers need to invent substantial amount of your time to pick and prepare the attack model and discovering zero-day vulnerabilities. Every one of these variables collectively provide an environment to launch targeted attacks [35].

2.3.3 Attack Phases

When cyber attackers strategize the way to infiltrate an organization's network and exfiltrate data, they follow a series of stages that comprise the attack lifecycle. The described cyber kill chain characterizes the progression of a cyber attack and in this 7 layer model, each one is critical. For attacks to successfully complete an attack, they need to progress through each stage. Blocking adversaries at any point within the cycle breaks the chain of attack.

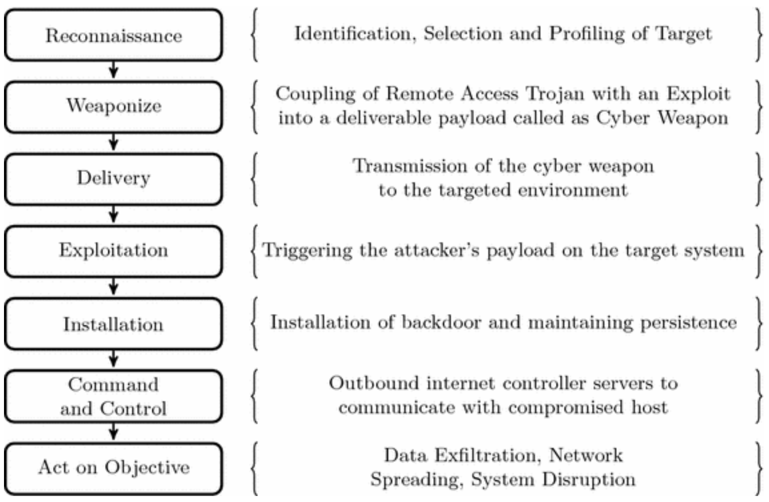

Reconnaissance
Identification, Selection and Profiling of Target
Weaponize
Coupling of Remote Access Trojan with an Exploit into a deliverable payload called as Cyber Weapon
Delivery
Transmission of the cyber weapon to the targeted environment
Exploitation
Triggering the attacker's payload on the target system
Installation
Installation of backdoor and maintaining persistence
Command and Control
Outbound internet controller servers to communicate with compromised host
Act on Objective
Data Exfiltration, Network Spreading, System Disruption

Figure 2.3: Phases of cyber kill chain (retrieved from article [38]).

The kill chain model mainly describes an Advanced Persistent Threat (APT), a classy attacker waging an organized attack campaign against a selected company. A targeted attack can be considered a super set of APTs [39]. Every stage is critical for ensuring the success of targeted attack.

Reconnaissance

Reconnaissance means gathering information about the potential target, which can affect individuals or organizations and is used in later stages to design and deliver the payload. No detail should be overlooked, no matter how innocuous it may seem. Recognition can be divided into target identification, selection, and profiling. For successful recognition, a strategy is needed. A typical strategy needs to include active and passive recognition.[40]:

- **Passively**: The attacker uses the vast amount of data available on on the internet, not iterating directly with the target and intrinsically, the target has no way of knowing, recording, or logging any activity.
- **Actively**: The attacker iterates directly with the target, involving deeper profiling and could trigger an alert to the target.

Weaponize

Weaponize means design a backdoor and a penetration plan, using the information gathered from reconnaissance. Attackers often use malware, commonly a Remote Access Trojan, or RAT, coupled with a delivery payload with a deliverable payload, such as an infected document, like a PDF [41].

Depending on the delivery method, weaponization can take many other forms, such as exploit kits. In this case, the attacker tricks the target to download the malware without hiding it. The target is encouraged to visit an infect website, which could result in a "drive-by download" of malware [41].

Delivery

Delivery is a critical part of the cyber kill chain, responsible for an efficient and effective cyber attack. In most of the cyber-attacks it is mandatory to have some kind of user interaction like downloading and executing malicious files or visiting malicious web pages. Delivery may be a high-risk task for an attacker because delivery leaves traces. While delivering the weapon multiple methods can be used to guarantee 100% success [35].

The Verizon [4] report classifies data breach through the following delivery mechanisms: Email Attachments, Phishing Attacks, Drive-by Download - The target is forced to download appealing malicious content from internet, USB/Removal Media, and DNS Cache Poisoning - Vulnerabilities in DNS are exploited to divert internet traffic from legitimate servers to attacker controlled destinations [42].

According to Internet Security Thread Report by Symantec, 65% of known APT groups used phishing emails for targeted attacks [43]. Email links account for nearly 40% of malware vectors, with email attachments account for about 18% [42].

[4] https://enterprise.verizon.com/

Exploitation

After delivering the weapon, is expected the target completes the required task on the target side. On execution, the next step is triggering the **exploite**, who silently install/execute the payload. To run the exploit, there are certain conditions that need to be met [35].

1. The user must use the software/operating system for which the exploit has been created.
2. The software/operating system should not be updated or upgraded to the versions wherein the exploit could not work.
3. Anti-viruses or any security system mechanism should not detect the exploit or payload neither in statistically nor dynamically scan during run time.

If all the conditions are fulfilled then exploit is triggered and can successfully install/execute the payload within the target's system. Exploits are made using vulnerabilities in software publicly identified as Common Vulnerabilities and Exposures (CVE) [5] [35].

Installation

The **installation** of a remote access trojan or backdoor on the victim's system allows the contender to maintain perseverance within the environment [44]. The common way to the computer would become infected by an infection vector, is for example, an infected removable media, which in turn will leave a malware executable in some unusual location. Eventually, the user will report this executable to an antivirus vendor, who in turn will analyze it and come up with a signature to detect it and maybe remove it. However, modern malware is not that simple. Nowadays, malware is multi-staged and depend on droppers and downloaders to deliver the malware modules. A **Dropper** is a program that will install and run the malware on the target system after trying to disabling security controls and hide the installed malware. A **Downloader** is similar to the Dropper but tended to be smaller than Dropper because they not contain malicious components. Instead, they connect to a remote repository and download the core components [35].

Installation life-cycle incorporates several resilience features in a way to maximize the success of the installation, protecting the attack entity.

Command and Control

Command and Control (C&C) system is employed to offer remote covert instructions to compromised machines. This enables for continuous connectivity for the environment and the detective measure activity on the defense. Can act as the place where the data can be exfiltrated. The architecture of C&C channel has evolved owing the exponential development of defensive mechanisms, namely anti viruses, firewalls, IDs, etc.

There are mainly three types of C&C communication structures, the traditionally centralized structure, the decentralized architecture and the most recent one, the Social Networks based structure [35].

[5] https://cve.mitre.org/

- The **Centralized Structure** refers to an only one server that command and control the infected machines. Its easy to manage. However, if the server is taking down will shutdown the entire C&C architecture.
- The **Decentralized Structure** the attackers use Peer-to-Peer (P2P) architecture for command and control. This type of architecture promotes scalability, fault tolerance, since redundant communication links can be formed to route information and P2P nature resolves the single point dependence of centralized architectures.
- The **Social Networks-Based Structure** is the most recent and dangerous one. For example, Facebook has 1.35 billion registered users as of third quarter of 2014. Most of the social network services are free and are considered benign for a big part of the world, becoming a viable option to attackers. High availability and reliable social networks are used to transmit information in a centralized/decentralized way to the infected machines. An example of such attacks is *Taidoor*.

Taidoor is a malware used for cyber espionage. In a typical attack, targets receive a spear-phishing email - attack via email or electronic communication, targeted at a specific individual, organization or company - which encourage them to open an attached file, and if opened on a vulnerable system, malware silently installed on the target's computer while a decoy document with a legitimate content is opened that is intended to alleviate any suspicions. *Taidoor* has been successfully compromising targets since 2008. This thread has evolved over time. Additionally, the well-known *Taidoor* network approach pattern has been modified, likely as a new way to avoid network-based detection [45].

Act on Objective

Once all the previous steps are completed, APTs finally begin to work on their main objectives. The command used by attacker depends on interest of attack [46]:

1. **Mass Attack**: The goal is to hit many targets as possible. Most of such attacks aims to getting bank, email or local system administrator credentials. An example of mass attack is the **BOTNets**. BOTNets are mainly used for Distributed Denial-of-Service (DDoS) attacks and virtual coin mining [35].
2. **Targeted Attack**: Its more sophisticated and carried with more caution. Most of the attacks are trying to get confidential data from target system. Spreading through the network also becomes the primary goal when the target is an organization [35]. These may include **data exfiltration, installing malware intend to disable or destroy systems, or forwarding to bigger systems linked to the system they have compromised** [41].

2.3.4 Data Exfiltration

Data exfiltration may be a major risk for many organizations, particularly those with highly valuable or sensitive information. Whether accidental or intentional, insider threats within a corporation could also be putting sensitive data in danger every day. Hackers also often target privileged insiders with credential theft attempts, including phishing and social engineering attacks. Also, transferring private information from the target system are often manual by someone with physical access to the computer or automated, administered through malware over a network [47].

Portuguese energy giant Energias de Portugal (EDP) was the latest company to be threatened with public disclosure of their sensitive data if they fail to pay the ransom. Hacking group called *Ragnarok*, known for using the custom Ragnar Locker ransomware that has been hitting managed service providers since late 2019, has publicly threatened to dump sensitive information from the 10TB of data they stole if the energy company does not opt to pay their ransom demand of $10.9 million [48].

Big data security is an understudied area. According to research from CA Technologies, databases are the amount one most vulnerable IT asset. If you think about it, databases are the crown jewel of sensitive information, and if an attacker accesses one, there's a high likelihood of uncovering something extraordinarily valuable to the organization. Cybercriminals can exploit vulnerabilities at the hardware, software and/or network layer level to secretly filter data stored on vulnerable systems, such as by IoT devices and the cloud. IoT devices are increasingly targeted, as these systems are sources of a large amount of data flow, including personal and confidential information from individuals, organizations and governments. Organizations, where an employee can bring their own device, may be potentially exposed to an increased risk of data exfiltration if those devices are used to access and store sensitive corporate information. Once a compromised device is successfully targeted and compromised, secret data exfiltration can easily occur [49]. In such a case, the intrusion can, in theory, be detected by identifying a channel that is being used to exfiltrate information.

Time aspects

An article developed by *Annarita Giani* et al. [50] makes an approach to the concept of timing covert channels. They first consider the bandwidth constraints of varied media, before moving on to a (rather) subjective evaluation of "covertness". Figure 2.4 shows the interval of your time required to exfiltrate a given amount of data (horizontal axis) for several different exfiltration methods (printing, burning DVDs, or a network transaction given a selected bandwidth constraint).

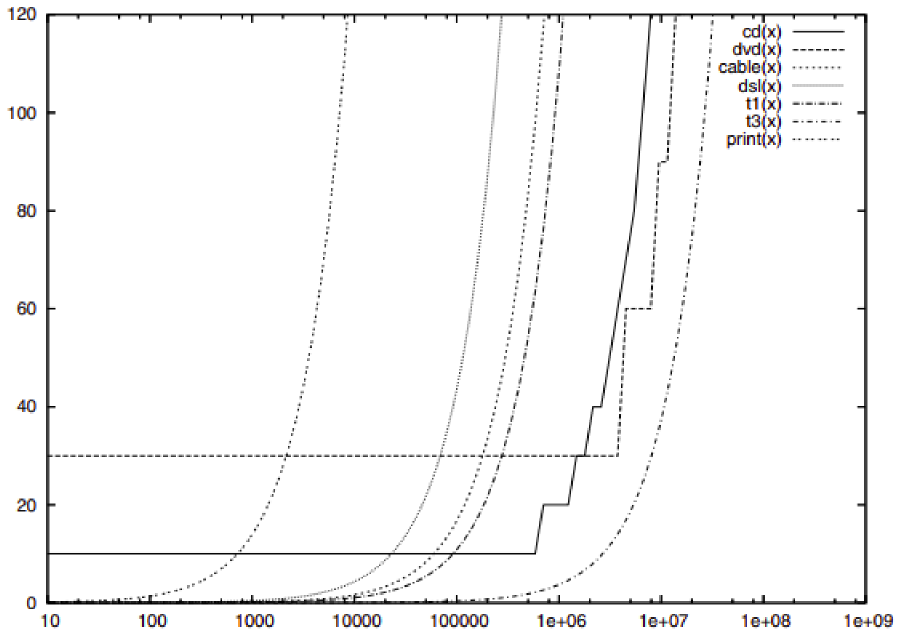


Figure 2.4: Exfiltration times (in minutes) given different amounts of data (in kilobytes) using various media. Note the log-scale on the horizontal axis. Moving 10KB could for instance be a letter, while 100GB could entail moving an entire database (retrieved from [50]).

For instance, for printing pages they assume that each page holds 3.6Kb in text and the printer can print a page in 3 seconds. For CD-ROM disks for data transfer, supposing that each disk contains 700 MB, they assume that the time to burn a CD is approximately 10 minutes. Then to transfer X MB we need at least $Y = \frac{X}{700}$ disks, taking a about $Y \cdot 10$ minutes. They must round up, since transferring only 10 kilobytes by CD-ROM still requires the use of a full CD-ROM (although the burning process will take significantly less time).

Assuming what was previously mentioned, informally, an operation is "more" covert if it's difficult to detect without the utilization of special tools that specifically search for it. In other words, they define covertness a characteristic of an operation which will be measured by the rate of usage of the media. If the apparatus is exploited at its maximum capacity, the operation is definitely visible with a covertness of zero; if instead it's exploited at a lesser rate, the operation are going to be increasingly covert. Therefore, to stay activity as covert as possible, the speed of usage, compared to the capacity of the equipment, should be kept as low as possible. The closer the capacity is to the rate at which the operation or transmission is executed, the more covert the transmission are going to be. Covertness is thus proportional to the difference between capacity and therefore the actual rate used:

$$Covertness \propto \text{(Capacity of the medium - Transmission Rate)}$$

The capacity and transmission isn't the sole condition to require a glance. For instance, document printing is less noticeable when the user uses a laptop in a workplace than in a public place. The stealthiness of the transmission, therefore, depends not only on its covertness but also the precise visibility of the operation; as an example, if a record of the method is kept during a log. Figure 2.5 gives an intuitive idea of how covert a given exfiltration method is compared to others.

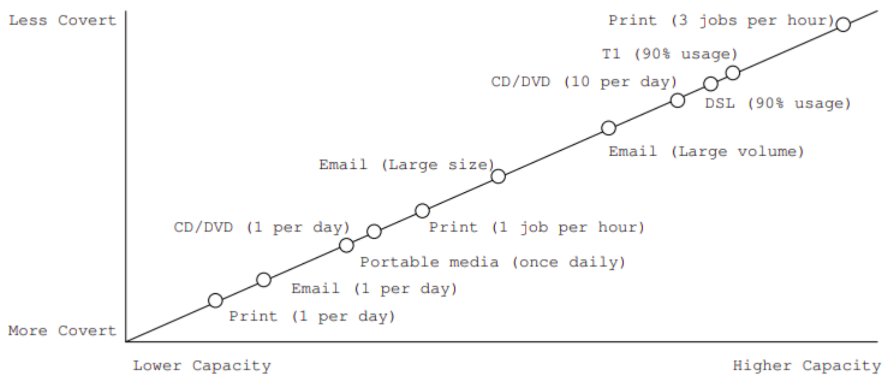


Figure 2.5: Intuitive covertness of the exfiltration methods given the rate at which the media is utilized (retrieved from [50]).

2.4 Monitoring

System monitoring provides a feature that aims to detect real attacks or attempted attacks on business systems and services. Monitoring depends heavily on the context, referring to the process of becoming aware of the system's state. Good monitoring is essential in order to effectively **detect attacks**, either originating from outside the organization or attacks as a result of deliberate or accidental user activity, **react to attacks**, a swift response is essential to stop the attack, and to minimize the impact or damage caused, and **must take into account the activity**, having a complete understanding of how systems are being used by users, allowing to ensure that systems are being used appropriately in accordance with organizational policies [51].

Monitoring can be done both at the protocol level and at the radio spectrum level. However, protocol monitoring is unable to monitor and is unaware of the existence of parallel channels. Considering this scenario, spectrum monitoring is an important ally. Spectrum monitoring helps spectrum regulators to plan and use frequencies, avoid incompatible use and identify sources of harmful interference. Spectrum monitoring is extremely necessary and important in the management of spectrum resources, radio stations and electromagnetic environments, providing valuable monitoring data, including spectrum occupation and signal characteristics, such as field strength, bandwidth, modulation type , location of emitters, etc [52].

As a result of technological developments, namely with the miniaturization of electronic components and the increase in processing capacity, it has enabled constant and rapid evolution of radio-communication and monitoring systems. New technologies in radio-communication systems may encompass adaptative frequency usage, co-frequency multiplexing, spread spectrum such the frequency hopping, among others. Software-defined radio is a typical example of a new radio-communication system. Future spectrum monitoring should have the capability for monitoring new radio-communication technologies and systems, such as detection of weak signals, con-frequency signal separation, RF sensor networking, and other technologies.

Two of the most common methods of monitoring, functional to both paradigms, both at the protocol level and at the radio spectrum level are active and passive monitoring.

- **Active Monitoring** simulates user behavior to determine potential network performance by "injecting" traffic into the environment. These tests can depend on what will be measured. Since test traffic mimics the service traffic, active testing is right for providing a real-time view of the end-to-end performance of a service, such things as latency (or delay), jitter (or delay variation), or packet loss [53]. There are a variety of tools used in active monitoring, such as the injection of Simple Network Management Protocol (SNMP) queries to obtain or even change settings on devices on the network environment, probing tools such as ping and traceroute to check for connectivity, and scanners, such as nmap, to scan a given network (e.g., check the IP addresses of active machines on a subnet).
- **Passive Monitoring** does not introduce traffic overhead in the network. On the other hand, this technique involves capturing and analyzing live network traffic, or traffic statistics. Large volumes of data are ideal for performing predictive analysis on intrusions and bandwidth usage baselines, commonly through machine learning strategies. At its simplest, passive monitoring could also be nothing more than the periodic collecting of port statistics, like byte and packet transmit and receive numbers [53].

In fact, the best of both worlds is the best option. Active and passive monitoring is necessary to obtain a complete image. Active monitoring should be used to provide real-time visibility into service level performance, Quality Of Service (QoS) and give visibility to possible network anomalies. Passive monitoring can be used to support active monitoring, first, it can be used for post-event analysis, such as identifying malicious traffic. When building a historical profile of traffic and signaling flows, the analysis can be performed to look for anomalous traffic. In addition, it can also be ideal for building a detailed understanding of customer usage patterns and application performance, enabling a deep understanding of root cause issues and customer behavior, allowing greater visibility into the customer experience [53].

Unfortunately, detailed information has a cost. Store every information can be unpractical due to the power and processing limitations. The required monitors to deal with traffic may be very expensive due to the requirements for processing and storing collected data. In addition, inferring typical user behavior and traffic characteristics can compromise and possibly violate user privacy through the monitoring system.

2.4.1 Data Collection

The main purpose of monitoring is to realize near real-time insight into the present state of the system. The extracted information helps to answer many information questions, assists in the verification of nonstandard behavior and shows more information about a reported issue.

The process of monitoring starts with gathering data by collection agents, specialized software programs running on monitored entities like hosts, databases, or network devices. Those agents capture meaningful system information, encapsulate it into data inputs, and then report these data inputs to the monitoring system at regular intervals. The inputs are then collated and aggregated into metrics to be presented as data points on a time series at a later stage. Input collection could also be a continuous process or it's going to occur periodically at even time intervals, depending on the nature of the measurement and therefore the cost of the resources involved in data collection. The data collection agents can be categorized into [54]:

- **White-box**
  - **Log parsers** : These extract specific information from log entries;
  - **Log scanners** : These count occurrences of strings in log files, defined by regular expressions;
  - **Interface readers** : These read and interpret system and device interfaces. Examples include readings of CPU utilization;
- **Black-box**
  - **Probers** : These run outside the monitored system and send requests to the system to check its response, such as *ping* or *HTTP* calls to a website to check their availability;
  - **Sniffers** : These monitor network interfaces, such Bluetooth interface, for instance, and analyze traffic statistics such as number of transmitted packets;

Before the data collection occurs, agents must be deployed into the monitored entities. However, in some circumstances, as the scenario that will later be referred to by this dissertation, it might be desirable to monitor remote entities without the use of deployable agents, referred to as *agentless data collection* in which the data is transmitted from monitored entity through an agreed protocol and is interpreted outside the monitored system [54].

Resort to agentless monitoring is an advantage in systems with heavy restrictions on custom software deployments, such as proprietary systems that disallow custom addictions or high-security systems policy restrictions imposed. Examples of *agentless data collection* include gathering statistics from entities running on networking gear via SNMP or periodically executing diagnostic commands via Secure Socket Shell (SSH).

2.4.2 Software-Defined Radio (SDR)

The term "Software Radio" was used in the early 1990's by *Joseph Mitola III* to refer to a type of radio that was able to implement different communication standards from the same hardware [55]. In a SDR, some of the radio functions typically implemented in hardware are converted into software like the signal modulation and encoding.

Today, with the advent of IoT and the resulting fragmentation of wireless technologies, they bring not only benefits, but also concerns and SDRs has a crucial role in monitoring IoT environments. In a practical way, e.g. in a given populated area, several individuals communicate with each other using various communication methods. Individuals use a panoply of devices for innocuous, day-to-day activities, however, there are a few malicious individuals (rogue agents) whose purpose is to cause harm and disrupt the lives of others. The communications between individuals are increasingly encrypted at various levels for privacy reasons. It's natural to assume that rogue agents choose to conceal their radio communications among the civilian (background) traffic while enjoying the privacy protection provided by encryption systems.

The set of communication technologies used by people in their day can reveal different transmission frequencies. The best solution to address the scenario of distinguishing different utilizations is to work with SDR, which will allow you to modify the radio frequency that is being analyzed and the way to decode the signal based on the device being evaluated. Therefore, it is not necessary to have different hardware for different devices, but a combination of single hardware and software that will allow changes to be made according to the requirements [56]. The arrival of SDRs offers an opportunity to identify dishonest agents by looking at radio communications in an area. SDR can be used as a flexible, low-cost scanner, with a focus on the statistical characteristics of communication standards.

Most articles and projects developed with SDR use RSSI values or characteristics taken directly from the RSSI values to train models. One of those projects was developed by *Vijay Bhuse* and *Ajay Gupta* [57] in which they propose methods to detect anomalies intrusions in wireless sensor networks. The basic idea is to reuse the system information already available that is generated in several layers of a network stack. They try to detect anomaly at multiple layers using the only existing system information. If the intruder escapes at one layer, there are still possibilities that it will be detected at other layers. The multi layer approach makes intrusion detection system robust. The problem is related to how they use RSSI values. They associate a neighbor with an estimated RSSI value and when nodes perform neighbor discovery, they record RSSI values for each neighbor. These recorded values are used to detect intrusion.

Another article developed by *Meng Zhang*, *Anand Raghunathan*, and *Niraj K. Jha* [58] proposed a medical security monitor that snoops on all the radio-frequency wireless communications to/from medical devices and uses multi layered anomaly detection to identify malicious transactions. They observe physical characteristics of the signal, such RSSI, time arrival, differential time arrival, and angle of arrival and generate limits and intervals considered normal. Thus, thresholds are used to demarcate boundaries between normal and abnormal values. An anomaly is detected if the signal allegedly sent by the device has unusually high or low strength. It is stated in this article that "... measurements are performed in an empty room with no object near the transmitter/receiver or blocking the transmission paths", obtaining good results. However, if the attacker stayed within the RSSI threshold, he would be able to access the devices. The solution presented to this problem is "... the monitor can analyze the packet contents and capture "smart" attacks that are not caught by physical anomaly detection". In conclusion, in spite of obtaining good results, the measurements were made in a "clean" space, with no real test. Therefore, if the monitoring system fails in real time, the answer will be to look at the content causing a breach of privacy and possibly data integrity.

In both articles, they are aware of the flexibility of RSSI according to external factors, however they put this characteristic aside and consider RSSI to be a constant variable, considering that its change is only affected by the distance between devices.

2.4.3 Intrusion Detection System (IDS)

Intrusion Detection System (IDS) are one of the tools that network administrators rely on to perform monitoring. An IDS would inspect network traffic with the help of agents on the network and sets off alarms when it detects suspicious activity. Essentially, the IDS would compare the traffic against what it considers normal traffic and, using a range of techniques, would generate an alarm. IDS can provide valuable information about anomalous network behavior. [59].

On the other hand, an IPS is very similar to an IDS, however, it can be configured to block potential threats by setting filtering criteria on network devices. Just as IDS monitors traffic in real-time by discarding any suspicious malicious packets, blocking traffic from malicious source addresses, resetting suspicious connections, or deploying an alarm [59]. In conclusion, an IPS system is proactive and, in theory, works autonomously, without the intervention of a network administrator, unlike an IDS.

False alarms are a huge problem for IDSs. For example, **false positives** may be generated by the IDS for legitimate hosts carrying out identical legitimate behavior that may appear malicious. Lets take an example, imagine a legitimate domain accessed frequently by hosts becomes temporarily unreachable. The failed Domain Name Service (DNS) queries to the same domain in this instance will be generated for many hosts and may appear suspicious, but should not be considered malicious activity. Likewise, a **false negative** would cause classifying malicious activity as benign.

An IDS can be an integral part of an organization's security, but they are just one aspect of many in a cohesive and safe system. It can be useful to monitor the network, but its usefulness depends on what the administrator does with the information, because **the detection tools do not block or solve potential problems**, they are ineffective to add a layer of security, unless the administrator has the right personnel and policy to manage them and act on any threads. Also, an IDS **cannot see into encrypted packets**, so attackers can use them to enter into the network. An IDS will not register these intrusions until the intrusion is discovered, and even if the packets have not been encrypted, the **information from an IP packet can still be spoofed**, and the IDSs can't see that [60].
IDSs can be classified in two categories [59]:

1. **Signature-based intrusion detection systems** compare monitored traffic against a database containing known threat signatures similar to virus scan software. The database need to be continually updated or will not detect new types of attacks. Signatures can be as simple as a source/destination IP address or contain many other protocol headers.
2. **Anomaly-based intrusion detection systems** use features of normal traffic to compare with the monitored traffic. The criterion of capturing normal traffic could be bandwidth usage, protocols, ports, arrival rate and burstiness [61]. Network behavior is usually defined by an automated training, while statistical modeling detects outliers with minimal false-positive rates. It is way more precise, efficient, and reliable than a

signature-based system, but they are sometimes used together to complement each other.

Host-based Intrusion Detection System (HIDS) is another point of view, because HIDS runs on individual hosts who monitoring for malicious behavior. A HIDS monitors and analyzes the internals of a computing system rather than its external interfaces [59]. One can think of a HIDS as an agent that monitors whether anything or anyone internal or external has circumvented the security policy that the operating system tries to enforce. Most virus scan software would have this feature where they also monitor inbound and outbound traffic in addition to the usual virus scanning. This can be helpful if the hosts have been compromised and form part of a bot to attack other networks/services. Unlike intrusion detection system that is deployed at strategic locations within the network to monitor inbound and outbound traffic to and from a domain area.

2.4.4 Network Anomaly Detection

Ideally, attacks should be detected as early as possible, or even predicted before they have started so that they can be prevented altogether. The spike in the total of network attacks, their severity, and complexity has forced administrators to use tools that rely on anomaly analysis to detect new and unforeseen phenomena, rather than solutions that look for traditional and well-known attacks.

Sabahi and Movaghar [62] developed a survey who categorized anomaly-based IDSs into the following five categories: (i) statistical methods; (ii) rule-based methods; (iii) distance-based methods; (iv) profiling methods and (v) model-based approaches.

- **Statistical methods** monitor the user or system behavior by measuring certain variables over time, e.g. the number of clicks on a page. This type of approach are used in HIDS, network-based IDSs, as well in application-based IDSs for detection malicious behaviors.
- **Distance-based methods** come in to try to overcome the limitations of the statistical approaches detecting outliers. They detect outliers by computing distances among points. However, it cannot handle data on substantial dimensional spaces.
- **Rule-based methods** predefines a set of rules to characterize normal behavior and then detects anomalous behavior as deviations from that set of rules.
- **Profiling methods** builds profiles of normal behaviors for different types of network traffic (for example, programs or users), and deviations from them are considered intrusions. Profiling methods vary widely, from different data mining techniques to various heuristic-based approaches.
- **Model-based approaches** builds models to characterize normal behavior and anomalies are detected as deviations from this model.

Time Aspects

In the same survey developed by Sabahi and Movaghar [62], they consider that IDSs can be distinguish into two main groups: (i) offline IDSs and (ii) real-time (online) IDSs. Offline

IDSs perform post-analysis of audit data. Offline analysis is often performed using static tools that analyze an environment snapshot looking for vulnerabilities and configuration errors.

Real-time IDSs attempt to detect intrusions in real-time or near real-time. They process data from data streams and analyze the data while the stream is in progress. Real-time IDSs would shoot an alarm when an attack is detected.

One aspect to take into account in real-time IDSs is *time granularity.* When data inputs are segmented into fixed intervals and summarized by a mathematical transformation in a meaningful way, they can be presented as a time series. The length of data intervals referred to as *time granularity*, depends on the types of measurements and the kind of information that will be extracted. One of the most important advantages of using time series for monitoring is their property of illustrating the process of change in the context of historical data. Fine-grained metrics tend to reveal a closer time to a phenomena and are useful for locating specific events. Although it can be more expensive to store. Coarse-grained metrics are much better suited to illustrate trends. Selecting the right granularity to present a metric is important for accurate data interpretation. Measurements of very small and very large granules can obscure the point you are trying to convey. A good granularity is important to answer the "one million dollars" question of monitoring: what has changed and when? [54]

2.4.5 IDS approaches using ML Techniques

The effectiveness of Machine Learning (ML) techniques in fraud detection, image recognition and text classification has encouraged security researchers to employ these algorithms for anomalous pattern detection and to identify abnormal behaviors to enhance the security in IoT networks [63]. The machine learning algorithms rely on learning data sets taken as inputs. For this reason, Machine Learning is being applied even in conventional methods of attack detection such as signature-based and anomaly-based in traditional internet networks [64].

Bhavani [65] and Ponthapalli [66] proposed an intrusion detection system **based on single machine learning classifiers**, Bhavani using random forest and decision trees techniques and Ponthapalli using decision tree, logistic regression, RF and Support Vector Machine (SVM) on KDD-NSL [6] dataset.

The researches showed that the intrusion detection system performs best with random forest classifier. They also discovered that the random forest classifier has the least execution time, but has the limitation of performing efficiently only with a single dataset. **Low detection as well as false positive rate were not solved** by the proposed work.

Marzia Z. and Chung-Horng L. [67] propose an implementation of an IDS **based on a set of multiple supervised and unsupervised machine learning algorithms that were aggregated using the voting classifier**. They adopted Kyoto2006 dataset [7]. The work increase the accuracy and performance of the current intrusion detection systems. This makes their work to attain a certain level of accuracy but **the recall of the result is low**

[6]https://www.unb.ca/cic/datasets/nsl.html

[7]http://www.takakura.com/Kyoto_data/

in some few cases which indicate high values of false negative rate.

Dutt I. [68] proposed a real-time hybrid intrusion detection in which a signature-based approach was used to detect well known attacks while the anomaly approach to detect novel attacks. In this work a **high detection rate was achieved** due to the fact that; patterns of intrusions that were able to escape the signature-based detection were able to be identified as attack by the anomaly detection technique. The model's **accuracy increased incrementally each day** up to a significant value of 92.65% on the last day of the experiment, also, as the model learns and trains the system each day, **the rate of false negative decreases** sharply. **The issue of slow detection rate persists when the model is applied on a very big size data**.

Some works being done on previous intrusion detection systems lack ability to work efficiently on different data sets. A work done by Zhou [69] shows that anomaly-based detection has room for improvement especially in false positive rate. The proposed work present a novel intrusion detection system that brings the benefit of combing ensemble classifier with feature selection, this provides an improved efficiency and high accuracy detection of intrusions. The work was carried out using three different data sets. For feature selection, CFS-BA based approach was used. The ensemble based approach increases the multiclass classification performance on unbalanced data sets.

Perez D. [70], proposed a hybrid network based intrusion detection system using multiple hybrid machine learning techniques. The supervised machine learning technique, Neural Network was combined with unsupervised machine learning, K-Means clustering with feature selection. Another combination was made consisting of SVM with K-means clustering. The results clearly showed that the **combination of such supervised and unsupervised machine learnings complement each other which increase the performance** of IDSs. The combination of SVM and K-means with feature selection returns the best accuracy.

The articles described, although few in number, present points that allow us to affirm that ensemble and hybrid classifiers have the better predictive accuracy and detection rate. Furthermore, the use of feature selection has to be considered in order to get rid of irrelevant, unwanted and redundant features to improve the efficiency and detection rate of IDS.

2.5 Machine Learning

The term *machine learning* refers to the automated detection of meaningful patterns in data. A few decades ago it has become a common tool in almost any task that requires information extraction from large data sets [71]. Since the success of a learning algorithm depends on the data used, ML is inherently related to data analysis and statistics. More generally, learning techniques are data-driven methods combining fundamental concepts in computer science with ideas from statistics, probability and optimization [72].

Recently, Machine learning was defined by Stanford University as "the science of getting computers to act without being explicitly programmed" [73]. Machine Learning is now responsible for some of the most significant advancements in technology, such as the new industry of self-driving vehicles like those developed by Tesla. ML models have become quite adaptative in continuously learning, which makes them increasingly accurate the longer they operate [73]. ML algorithms combined with new computing technologies promote scalability and improve efficiency resolving a variety of organizational complexities. Although, it cannot always provide a correct analysis or cannot always provide an accurate result based on the analysis, but it gives a predictive model based on historical data to make decisions. The more data, the more the result-oriented predictions that can be made.

2.5.1 Machine Learning Process

Part of this dissertation can be described as a machine learning process from gathering data until the development of a model with a good performance and generalization, therefore, it is important to refer to the entire machine learning lifecycle.

Data is at the core of any application of machine learning. Among the many challenges in machine learning, **data collection** is becoming one of the critical bottlenecks and is the first step in the machine learning process. It is known that the majority of the time for running machine learning is spent on preparing the data, which includes collecting, cleaning, analyzing, visualizing, and feature engineering. This is one of the most important steps since a good result is not directly related to a good implementation of the algorithm, but to the quality and quantity of the data. In addiction, it is not possible to say in advance how much data is needed because everything depends on the algorithm that will be used. Only after testing is it possible to find out if the estimates are suffering from high polarization or variation [74].

Real-world data is never perfect. The data collected from multiple sources can be of a heterogeneous nature, containing duplicate data or missing data, or a lot of unnecessary extra data and therefore **pre-processing** [75], [76] may be necessary to produce consistent data sets for training purposes and verification. The data must be refined to make it suitable for ML. Pre-processing can also seek to reduce the complexity of the data collected or design resources to assist in training [77], [78]. Once refined and formatted, data can be summarized in a set of instances with multiple features. Each feature is a measurable individual property of a phenomenon that is being observed. **Data visualization** could help to see if there are any relevant relationships and how it's possible to take advantage, and as well show if there are any data imbalances present.

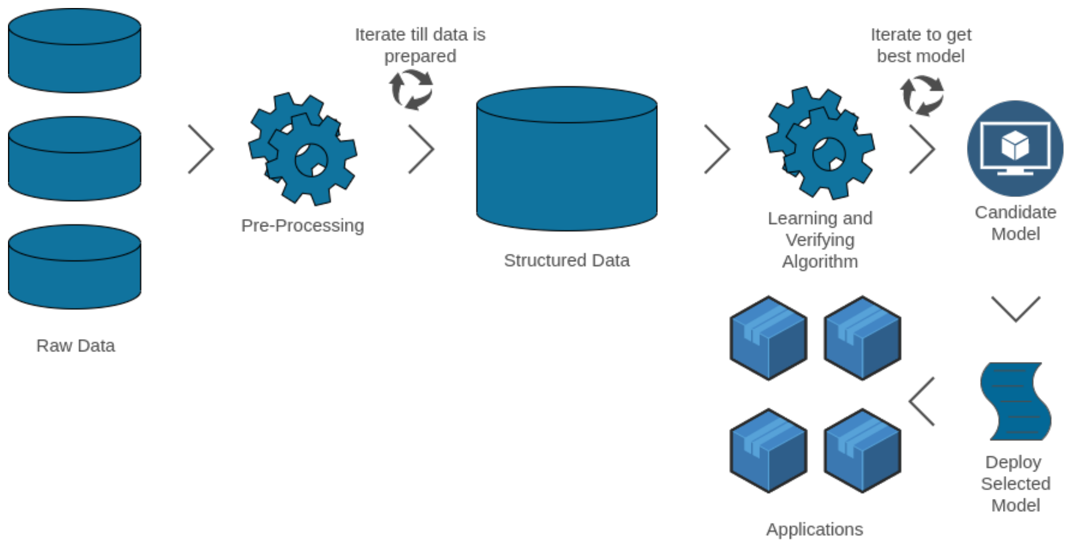

Iterate till data is prepared
Iterate to get best model
Pre-Processing
Structured Data
Learning and Verifying Algorithm
Candidate Model
Raw Data
Deploy Selected Model
Applications

Figure 2.6: Machine Learning Process.

The next step starts by **selecting the type of model** to be produced. This model selection is undertaken with reference to the problem type (e.g., classification or regression), the volume and structure of the training data [79], and often in light of personal experience. When the resulting ML model achieves satisfactory levels of performance, the next stage of the ML worklow is **validate the model**. Otherwise, the process is repeated, where additional data are collected, pre-processed and analyzed in order to improve the training further. The challenge of this stage is to ensure that the trained model performs well on new, previously unseen inputs, i.e. it has a good generalization. Assuming that the verification result contains all the required assurance evidence, a system that uses the now verified model is assembled.

2.5.2 Feature Dimensionality

Machine learning can be used on different tasks that are often characterized by a high dimensional space of features. The number of features for a dataset is referred to as its dimensionality. More input features often make a predictive modeling task more challenging to model, more generally referred to as the curse of dimensionality. Adding features without increasing the number of training samples as well, the dimensionality grows and becomes sparser and sparser. In fact, this brings a problem, the more dimensions the data has, the easier it is to find a hyperplane separating categories in the training data; but at the same time, the harder it gets to also perform well on the unseen data (for example, test data). The reason is that because we have more dimensions that we can choose from to lay the hyperplane through, we are much more prone to overfitting. Overfitting happens when the model corresponds too closely to a particular set of data and doesn't generalize well. An overfitted model would work very well on the training dataset, but will fail on the unseen dataset and became the prediction unreliable.

Moreover, features used to describe learning tasks are not necessarily all relevant and beneficial for the inductive learning task. In addition to the fact that a large number of input resources can degrade the performance of machine learning algorithms as stated above, it can even slow down the induction process, providing results similar to those obtained with a much smaller subset of resources. Therefore, it is often desirable to reduce the number of input resources [80].

Feature Selection

In a machine learning configuration, a matter of great interest is to estimate the influence of a given input feature on the forecast made by a model. Understanding which features are important helps to improve our models, increases confidence in model prediction and isolates undesirable behaviors.

To remove an irrelevant resource, a feature selection criterion is required to measure the relevance of each feature based on the output labels. From the point of view of machine learning, if a system uses irrelevant variables, it will use that information for new data leading to poor generalization.

The best subset contains the least number of dimensions that most contribute to accuracy, discarding the remaining unimportant dimensions since the main aim of feature selection is to determinate a minimal feature subset from a problem domain while retaining a suitably high accuracy in representing the original features.

The feature selection has an impact at several levels, from the **hardware level** reducing the limitation of storage requirements and has immediate effects on data analysis tasks by accelerating the execution time of the learning algorithms, at the **data level**, since it improves the quality of the data, removes the redundant, irrelevant or noisy data, reducing the overfitting, less redundant data means less opportunity to make decisions based on noise and allows a better understanding of them and a possible visualization and at the **level of the learning algorithm**, improving performance by gaining predictive precision and increasing the speed of the algorithm [81].

Generally speaking, feature selection methods can be divided into three main categories:

- **Filter Methods** : These methods select features based on discriminating criteria that are relatively independent of any machine learning algorithm. Instead, features are selected on the basis of their scores in various statistical tests for their correlation with the outcome variable. Recently, have been proposed several methods to select features with minimum redundancy. The methods use a Minimum Redundancy-Maximum Relevance (MRMR) feature selection framework. The point is to choose features that are maximally dissimilar, to expands the representative power of the feature set and improves their generalization properties [81].
- **Wrapper Methods** : These methods work by evaluating a subset of resources using a machine learning algorithm that employs a research strategy to examine the space of possible subsets of resources, evaluating each subset based on the quality of performance. The problem is essentially reduced to a research problem since they aim to find the

best possible combination of features that result in the best performant model. These methods are usually very computationally expensive [82].

- **Embedded Methods** : These methods combine the qualities of filter and wrapper methods. The embedded methods complete the feature selection during the model training, which is why we call them embedded methods. A learning algorithm takes advantage of its own variable selection process and performs feature selection and classification at the same time. The embedded methods solve both issues from filter and wrapper methods by combining their advantages, taking into consideration the interaction of features like wrapper does, but also faster like filter do, although more accurate than filter methods and find the feature subset for the algorithm being trained [83].

For any application, several feature selection algorithms can be applied and the best one can be selected that meets the required criteria. **A problem is the stability of the feature selection algorithms.** The stability of a feature selection algorithm can be viewed as the consistency of an algorithm to produce a consistent feature subset when new training samples are added or when some training samples are removed. If the algorithm produces a different subset for any perturbations in the training data, then that algorithm becomes unreliable for feature selection. In the multi-criteria fusion algorithm developed by Feng Yang and K. Z. Mao. [84] are used **multiple feature selection algorithms to classify/score** the features. They are combined to obtain a robust subset based on the combination of multiple classifiers to improve accuracy.

PCA

Feature selection and feature projection seek to reduce the number of attributes in a data set, the former removes the attributes, while the latter creates a new combination of attributes, but unlike feature selection, feature projection needs to ensure that it conveys similar information concisely, like the PCA [85].

Generally, PCA can be formulated as a mapping from an original feature space to an appropriate subspace such that a learning criterion is optimized. PCA enables us to visualize high dimensional feature vectors in a low dimension and analyze the distribution of the reduced feature vectors. As a result, we can select a classifier that has the best performance for the reduced feature vectors.

Introduced by *Karl Pearson*, in 1901, the data in a higher-dimensional space need to map to a lower space, although, the variance of the data in the lower dimensional space should be maximum [86]. The total amount of variance in PCA is equal to the number of observed variables being analyzed. In PCA, observed variables are standardized, e.g., mean=0, standard deviation=1 and diagonals of the matrix are equal to 1.

The number of components extracted is equal to the number of observed variables in the analysis. The first principal component identified accounts for most of the variance in the data. The second component identified accounts for the second largest amount of variance

in the data and is uncorrelated with the first principal component and so on. Components account for maximal variance are retained while other components accounting for a trivial amount of variance are not retained.
Eigenvalues indicate the amount of variance explained by each component, eigenvectors are the weights used to calculate components scores [87].

Let's say, there is a dataset which is $d + 1$ dimensional. Where d could be thought as X_{train} and 1 could be thought as y_{train} (labels) in modern machine learning paradigm. So, $X_{train} + y_{train}$ makes up our complete train dataset. Also, let's assume we are left with a three-dimensional dataset after ignoring the labels i.e $d = 3$.

For PCA to work properly, you have to subtract the mean from each of the data dimensions. The mean subtracted is the average across each dimension. So, all the x values have $\bar{x}$ (the mean of the x values of all the data points) subtracted, and all the y values have $\bar{y}$ subtracted from them. This produces a data set whose mean is zero.

After this, is computed the covariance matrix of the whole dataset. Recall that covariance is always measured between 2 dimensions. If we have a data set with more than 2 dimensions, there is more than one covariance measurement that can be calculated. For example, from a 3 dimensional data set (dimensions X, Y, Z) we could calculate $cov(X,Y)$, $cov(X,Z)$ and $cov(Y,Z)$. In fact, for an n-dimensional dataset, we can calculate $\frac{n!}{2(n-2)!}$ different covariance values.

$$cov(X,Y) = \frac{1}{n-1}\sum_{i=1}^{n}(X_i - \bar{x})(Y_i - \bar{y}) \tag{2.1}$$

So, since the non-diagonal elements in this covariance matrix are positive, we should expect that both the X and Y variable increase together. Now, we can easily compute eigenvalue and eigenvectors from the covariance matrix that we have above. To do this, we find the values of λ which satisfy the characteristic equation of the matrix A, namely those values of λ for which

$$det(A - \lambda I) = 0 \tag{2.2}$$

where I is the N×N identity matrix, where N is the dimension of the dataset. Once the eigenvalues of a matrix A have been found, we can find the eigenvectors. For each eigenvalue λ, we have

$$(A - \lambda I)x = 0 \tag{2.3}$$

where x is the eigenvector associated with eigenvalue λ. So, by this process of taking the eigenvectors of the covariance matrix, we have been able to extract lines that characterize the data. The rest of the steps involve transforming the data so that it is expressed in terms of them lines. And now, is where the notion of data compression and reduced dimensionality comes into it. It turns out that the eigenvector with the highest eigenvalue is the principle component of the data set.

In general, once eigenvectors are found from the covariance matrix, the next step is to order them by eigenvalue, highest to lowest. This gives you the components in order of significance. Now, if you like, you can decide to ignore the components of lesser significance. You do lose some information, but if the eigenvalues are small, you don't lose much. If you leave out some components, the final data set will have less dimensions than the original. To be precise, if you originally have n dimensions in data, and it's calculated n eigenvectors and eigenvalues, and then are chosen only the first p eigenvectors, then the final data set has only p dimensions [88].

Therefore, using the main principal components can reduce a dataset with high dimensions and massive inter-correlated variables to a few features with primary characteristics. This will help avoiding hypothetical and uncorrelated components (e.g. noise), which could interfere with the extraction of information [89].

Tree-based Classifiers

A standard approach is to use a classifier that differentiates all categories. However, training multiple classifiers will be very expensive and needs a common set of resources for all categories. Another option is to separate the data set into smaller classification groups, subsets, each dealing with its own set of features. This allows the use of a specific set of options for each set of categories. Splitting the problem can improve accuracy [90], and in some scenarios, the use of a smaller classifier can improve generalization [91]. Tree-based models can be very adaptable to the data and have low bias. A low-bias model are often highly flexible and has the capacity to suit a spread of various shapes and patterns, however in order to achieve low bias, models tend to demonstrate high variance. For example, let's consider an easy sequence of data points like a daily stock price. A moving average model would estimate the stock price on a given day by the average of the data points within a particular window of the day. The dimensions of the window can modulate the variance and bias here. For a little window, the average is far more aware of the data and has a high potential to match the underlying trend. However, it also inherits a high degree of sensitivity to those data within the window, and this increases variance. Widening the window will average more points and can reduce the variance within the model but also will desensitize the model fit potential by risking over-smoothing the data (and thus increasing bias).

There are a number of common methods for dividing a multi-class problem. Trees are intuitively a good choice because they require the training of fewer base classifiers than other techniques and are much faster in classifying new points. Feature selection using Tree-based Classifiers comes under the category of Embedded methods. Common feature set selection techniques can be grouped into several categories, namely, ranked selection, basic search, and sequential search [92].

- **Ranked** feature selection techniques simply apply a performance measure to every feature and choose the K number of features with the very best score or features with a worth over a particular threshold.

- **Basic search-based** feature selection techniques is a brute force method that includes techniques like exhaustive search. However, it is to exhaustively evaluate all possible combinations of the input features, then find the most effective subset. Obviously, the exhaustive search's computational cost is prohibitively high, with a substantial danger of overfitting. An overfitting scenario occurs when, in the training data, the model has an excellent performance, however when we use the test data the result is bad. Overfitting can have many causes and usually is a combination of **not enough data** and get more data can sometimes fix overfitting problems or **too many features**. Having a lot of features is like having a lot of dimensions. Effectively, it means that the data is more sparse, so it is much more likely to reach a conclusion that is not guaranteed.
- **Sequential search** techniques create a feature set by incrementally adding or removing candidate features from the feature set. The most popular sequential search techniques are the *sequential backward selection* method as the search algorithm was introduced already by Marill and Green (1963) and its "bottom up" counterpart known as *sequential forward selection* by Whitney (1971) [93]. Both these methods are generally sub-optimal and suffer from the "nesting effect", which means that features that are eliminated will not be considered for other iterations.

2.5.3 One-Class Classification

The traditional multi-class classification paradigm aims to classify an unknown data object into one of several predefined categories. The problem appears when an unknown data object does not belong to any of those categories. There are scenarios that classification task is just not to allocate a test object into predefined categories but to decide if it belongs to a particular class or not.

In OCC, one of the classes (which we will arbitrarily refer to as the target class) is well characterized by instances in the training data, while the other class (outlier class) has either no instances or very few of them, or they do not form a statistically-representative sample of the negative concept [94].

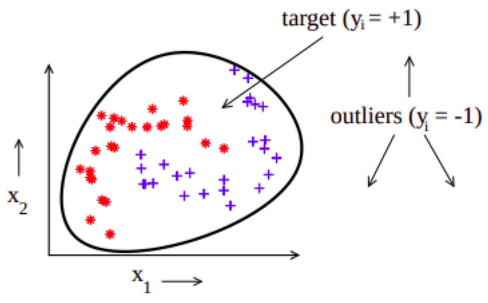


Figure 2.7: Scatter plot of the traditional one-class classification problem (retrieved from [95]).

In the literature a large number of different terms have been used for this problem. The term one-class classification had his first mention from Maya [96], but also outlier detection [97] or novelty detection [98] are used. The first application for data description and still used today is the detection of anomalies, for the detection of objects that do not resemble the data set in any way. This branch of machine learning technology has not only gained tremendous fame in classical domains like banking, financial institutions, medical and pharmaceutical science and telecommunication but also emerged as a key player in many IoT based applications for monitoring and predicting system failures [99].

The anomalies (outliers) are caused by human error, instrument error, natural deviation in populations, fraudulent behavior, unexpected changes in behavior, or faults in the system. The knowledge of the anomalies in data has a huge impact on the selection of a suitable approach for developing the detection system. As Pierre Lafaye de Micheaux, a Canadian/French/Swiss statistician, says *"Outliers are not necessarily a bad thing. These are just observations that are not following the same pattern as the other ones. But it can be the case that an outlier is very interesting. For example, if in a biological experiment, a rat is not dead whereas all others are, then it would be very interesting to understand why. This could lead to new scientific discoveries. So, it is important to detect outliers."* [99].

There are three approaches to face the problem on the level of modeling a strategy that could be considered based on the nature of data.

- **Supervised Learning** : This approach requires pre-labeled data tagged as normal or abnormal (or even specific known types of abnormal behaviors). One standard formulation of the supervised learning task is the classification problem: the model makes use of a function which maps a vector into one of the several classes by looking for the input-output examples of the function [100].
- **Unsupervised Learning** : This approach is using data sets where the output labels are not provided. Hence, instead of trying to predict a particular output for each input, these algorithms attempt to discover the underlying structure of the input data, grouping similar inputs together. This approach assumes the data has a static distribution which can be described by statistical models and flags the data points having values, not within the distribution as outliers. An example of a simple unsupervised learning algorithm is k-nearest neighbor clustering.
- **Semi-supervised Learning** : This approach focus on modeling only normality, requiring either pre-classified data marked as normal or assuming that the training only contains normal data, only applicable to cases where abnormalities are occasional events. For these algorithms, the normal pattern is learned by a supervised model and are apply an unsupervised method to induce the boundary of normality. This approach can be the most favorable for cases in which normal data is highly available but it's very hard to obtain abnormal data, such as an anomaly detection system.

The problem of this dissertation can be described as a classification problem where the classes are sampled very well, while the other class is severely undersampled. The measurements of the undersampled class might be very difficult to obtain. For instance, in a machine monitoring system where the current condition of a machine is examined, an alarm is raised when the machine shows a problem. Measurements on the normal working conditions of a machine are very cheap and easy to obtain. On the other hand, measurements of outliers would require to generate all faulty situations, which are impossible. Only a method trained on just the target data can solve the monitoring problem.

In this dissertation, we will focus on solutions of one-class classification with the creation of artificial outliers for the evaluation of the different models though. For classifying a class, One-Class Support Vector Machine (OC-SVM), Isolation Forest (IF), Local Outlier Factor (LOF), Kernel Density Estimation (KDE), and Gaussian Mixture Model (GMM) have been proposed to test for anomaly detection.

One-Class Support Vector Machines

SVM are today a very popular machine learning technique that can be used in a variety of applications, including, for example, handwritten digit recognition, object recognition, text categorization, and also anomaly detection. In those utilizations, SVM performs at least as good as others methods in terms of the generalization error [101], because they take the capacity of the model into account, which is the flexibility of the learned model to represent any training dataset with a minimal error.

On the other hand, OC-SVM attempt to determine a decision boundary that obtains the maximum separation between the points and the origin [102]. According to *Mennatallah Amer*, *Markus Goldstein* and *Slim Abdannadher*, the idea of getting the maximum separation between the points and the origin was hindered by the inability to learn non-linear decision boundaries as well as the inability to account for outliers. Although, both problems were solved by the introduction of kernels and the incorporation of soft margins [103]. According to *Yanxin Wang*, *Johnny Wong*, and *Andrew Miner* [104] the main idea is that the algorithm maps the data into a feature space H using a appropriate kernel function, and then attempts to determinate the hyperplane that separate the mapped vectors from the origin with maximum margin.

For example, given a data set $(x1, y1), (x2, y2), ... \in R^n \cdot \{\pm 1\}$, let $\phi : R^n \to H$ be a kernel map who transforms the training dataset into a feature space H. Then, to divide the dataset from the origin, we need to solve the bellow quadratic programming equation:

$$\min \left(\frac{1}{2} \|w\|^2 + \frac{1}{\upsilon \cdot l} \sum_{i=1}^{t} \xi_i - \rho \right)$$

subject to

$$y_i(w \cdot \Phi(x_i) \geq p - \xi_i, \xi_i \geq 0, i = 1, \ldots, l$$

where $\upsilon \in (0, 1)$ is a parameter that controls the trade off between maximizing the distance from the origin and containing most of the data in the region created by the hyperplane and is the ratio of "outliers" in the training dataset. Then the decision function

$$f(x) = sign((w \cdot \Phi(x)) - \rho)$$

will be positive for most samples x_i contained in training set. The data will be linear divided by a hyperplane, however, some data sets are not linearly separable. Thus, kernel functions, such as polynomial kernel and radial basis kernel, are widely used in SVMs to map from the original data to a higher dimension feature space in order to make data set linearly separable [105].

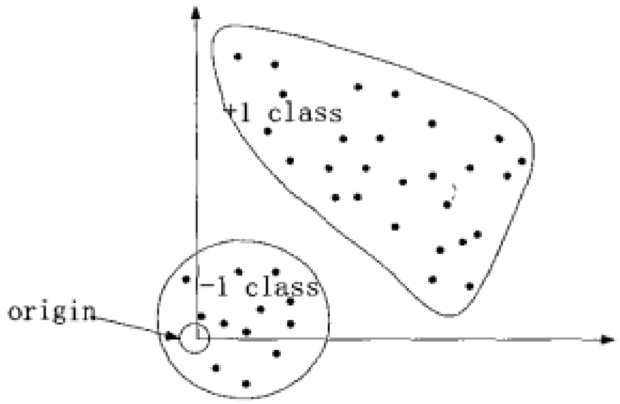


Figure 2.8: Geometry interpretation of one-class SVM based classifier (retrieved from [106]).

Isolation Forest

Another approach proposed for this dissertation is a model-based method that explicitly isolates anomalies rather than profiles normal instances. For this, the method proposed take advantage of two quantitative properties, they are the minority consisting of fewer instances and they have attribute-values that are very different from those of normal instances. In short, they are few and different, which make them more susceptible to isolation than normal points. The proposed method, IF, builds an ensemble of trees for a data set, then anomalies are those samples which have short average path lengths on the trees.

According to *Fei Tony Liu*, *Kai Ming Ting*, and *Zhi-Hua Zhou*, the difference between isolation and profiling, is the isolation characteristic of trees enables them to build partial models and exploit sub-sampling to an extent that is not feasible in other methods. Another different is the isolation forests utilizes no distance or density to detect anomalies, and this eliminates the major computational cost of distance calculation in all distance-based methods and density-based methods. In terms of complexity, isolation forest has a linear time complexity with a low constant and a low memory requirement. Isolation forests has also the capacity to scale up to handle extremely large data size and high dimensional problems with a large number of irrelevant attributes [107].

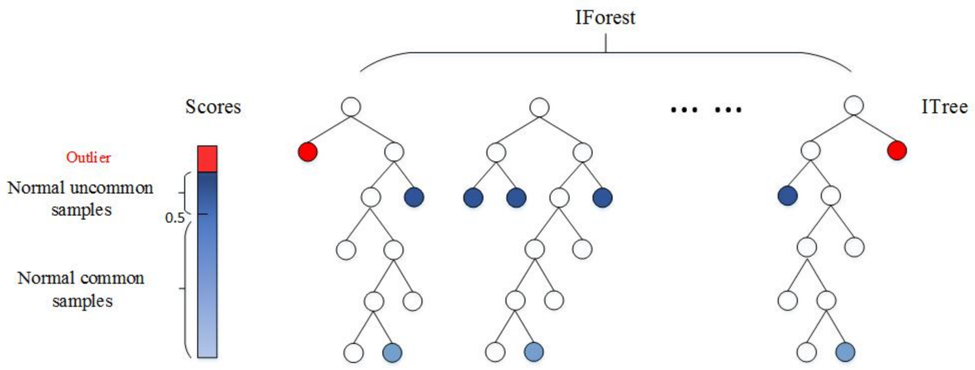


Figure 2.9: Representative subset selection and outlier detection via isolation forest (retrieved from [108]).

Local Outlier Factor

The LOF is an unsupervised technique first proposed for finding outliers in multidimensional data sets. The value of the local outlier factor indicates the extent to which a sample is an outlier [109].

In many applications, different portions of a data set can exhibit very different characteristics, and it is more meaningful to decide on the possibility of an object an outlier based on other objects in its neighborhood. In the LOF algorithm, the difference in density between a data object and its neighborhood is the degree of being an outlier, known as its *local outlier factor* [110]. The outliers are the data objects with high LOF values, whereas data objects with low LOF values are likely to be normal with respect to their neighborhood. In other words, a high LOF value is an indication of a low density neighborhood, and hence, a potential outlier. The computational procedure of LOF can be found in [111].

Unfortunately, the LOF algorithm's time complexity is $O(n^2)$, where n is the data size. It is designed to compute the LOF for all objects in the data set, which results in a computationally intensive process, since it requires a large number of k-Nearest Neighbours (kNN). Because of this issue, designing efficient and reliable intrusion detection systems based on the LOF method is challenging [110].

However, LOF has been shown to perform well in detecting abnormal behavior in a network IDS [112]. LOF has been used to identify several novel and previously unseen intrusions in real network data that could not be detected using other intrusion detection systems such as SNORT [8].

Kernel Density Estimation

An alternative approach to detecting statistical anomalies is the use of KDE. Kernel density estimation is a non-parametric technique to estimate probability density functions. Once it is non-parametric, it makes no assumptions about the form of the underlying distribution, allowing it to naturally follow the shape of the training data. When using kernel density

[8]https://www.snort.org/

estimates to detect anomalies, one sets a minimum probability threshold; if the kernel density estimate yields a probability that is below the threshold, anomaly detection is declared. The probability threshold can be set so that a certain small percentage of the historical data would be considered abnormal.

KDE is a technique dating back to the sixties [113]. Due to its computational complexity, however, it did not gain significant attention until the early nineties due to the increased availability of computational power [114]. Even with added computational power, kernel density estimation is not highly used in higher dimensions due to the curse of dimensionality [115] which hinders its ability to accurately measure the distribution. However, dimensionality is not an issue if dimension reduction techniques are used beforehand.

Gaussian Mixture Model

Another possible approach is to use a **Density Method** which directly estimate the density of the training data [116] and to set a threshold on this density. Several distributions can be assumed, such as a Gaussian or a Poisson distribution, and numerous tests, called disagreement tests, are then available to test new objects. In this thesis we will consider the **Gaussian mixture model** [117].

A GMM is a parametric probability density function represented as a weighted sum of Gaussian component densities. GMM parameters are estimated from training data using the iterative **Expectation-Maximization (EM)** algorithm from a well-trained prior model.

When the sample size is sufficiently high and a flexible density model is used, this approach works very well. However, it requires a large number of training samples to overcome the curse of dimensionality. If the dimensionality of the data and the complexity of the density model are restricted, this can be avoided, but then a large bias may be introduced when the model does not fit the data very well. When a good probability model is assumed and the sample size is sufficient, this approach has a very big advantage. When one threshold is optimized, automatically a minimum volume is found for the given probability density model [95].

CHAPTER 3

Scenario and Methodologies for Anomaly Detection

As stated throughout chapter 2, with the advent of IoT and wireless mesh networks, security risks inherent to these types of networks have grown in number, and severity, especially when attackers use network resources in an illicit manner, or perform data exfiltration. Most of the approaches current available for anomaly detection perform packet inspection, which may inadvertently reveal the private behavioral patterns of its users. Moreover, in situations where the communications is encrypted, these strategies could be ineffective. Physical layers even though more challenging to differentiate individual patterns of users/nodes, there is no way to identify private information, as used in layers 2 and 3.

As stated throughout this document, the objective of this dissertation is to develop a framework for detecting anomalies, measuring activity and silence periods and active and inactive frequencies in a given network and training OCC models.

3.1 Case scenario: Data exfiltration via Bluetooth from a Rogue Device

In the figure 3.1, the basic idea of using IoT devices in a business network context is presented, putting at the outset that at least one of them contains a bluetooth interface. For the development of this proof of concept, it is placed at the outset that the presented network is compromised and the attacker in question is ready to exfiltrate important data, jeopardizing the normal flow of the company.

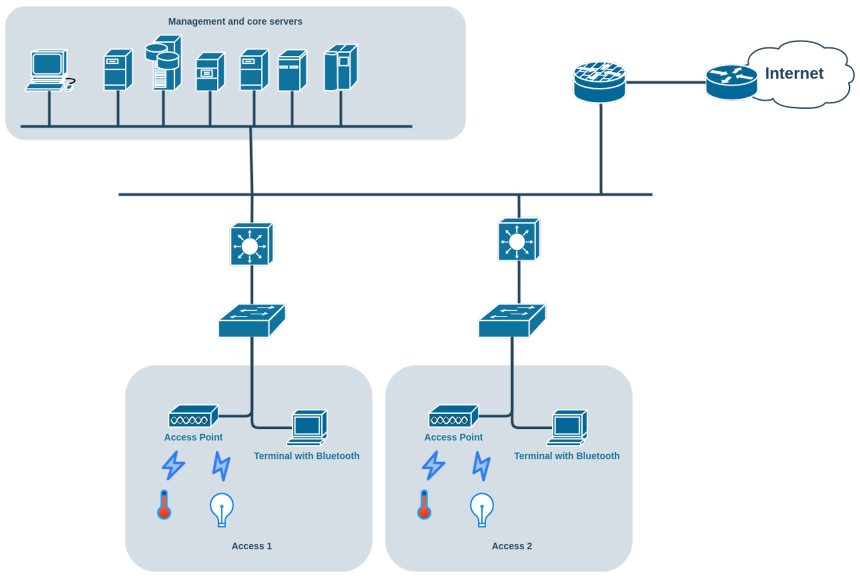


Figure 3.1: Basic idea of an IoT network

This attacker (figure 3.2) already has access to the internet and access to the machine he has committed to access the company's network. However, in order not to raise suspicions, instead of filtering the data through the network's gateway, where it contains systems that will alert and prevent the information from leaving, it groups all the information on the device that compromised and exfiltrates the data through Bluetooth.

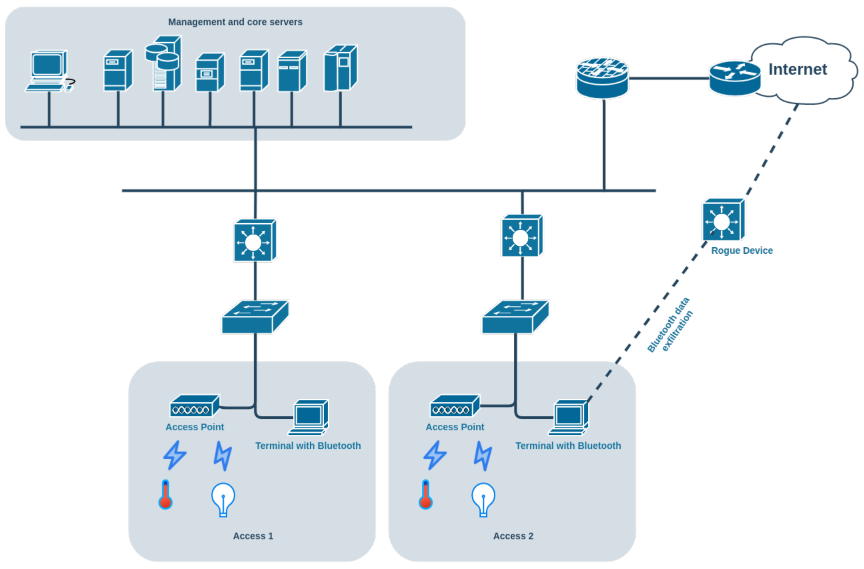


Figure 3.2: Compromised network

What this dissertation proposes is a framework for detecting anomalies that, as mentioned previously, monitors the physical layer, being a gap in existing technologies and recent studies. The structure (figure 3.3) would be composed of several devices installed in various positions due to the low range of Bluetooth, in which they would report anomalies to a central entity through a network parallel to the company's network.

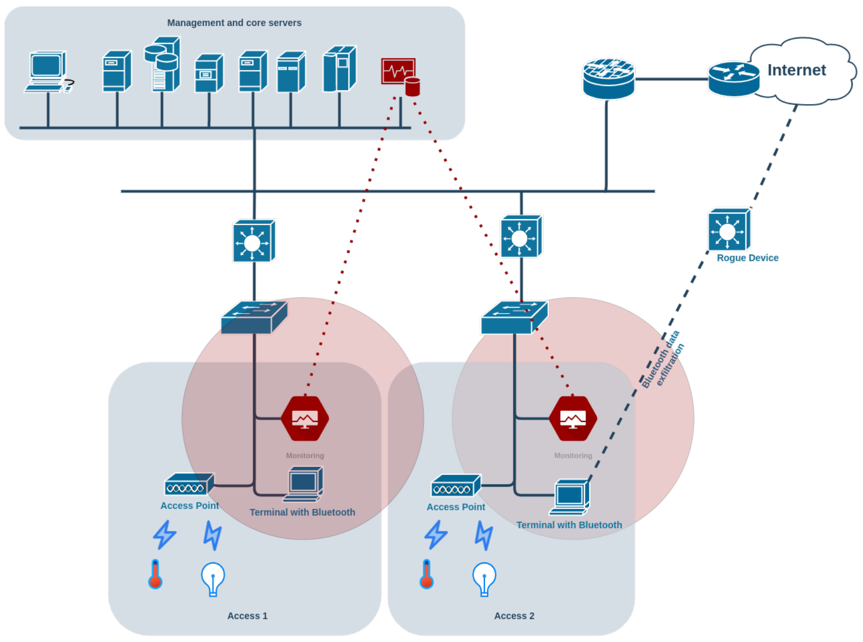


Figure 3.3: Compromised network with a framework for detecting anomalies on the physical layer

3.2 System Overview

In order to facilitate the visualization of the entire system, it will be separated in steps, and detailed by the figure below (figure 3.4).

To provide an appropriate solution in network anomaly detection, we need the concept of normality. The idea of normal is introduced by a model that expresses relations among the fundamental variables involved in system dynamics. Consequently, an event or an object is detected as anomalous if its degree of deviation with respect to the profile or behavior of the system, specified by the normality model, is high enough.

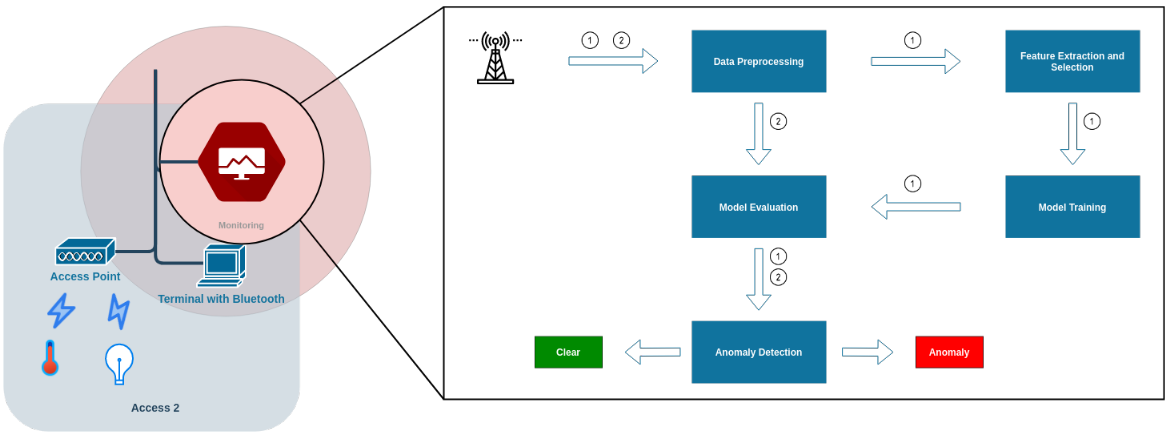


Figure 3.4: Overall System Overview Diagram

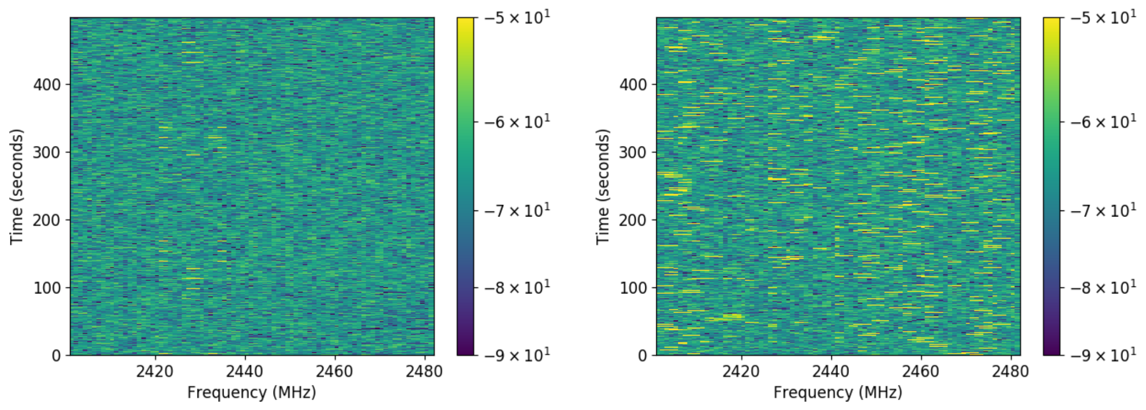
Therefore, in a practical way, the sniffer records power indicators RSSI from the processing of Fast Fourier Transform (FFT), which is transformed into a binary activity timeline. From it, a series of statistical characteristics that model normal behavior are extracted and, finally, anomaly detection is evaluated.

The system has mainly two modules: (1) a modeling module and (2) a detection module. One trains the systems to get the normality model M. The obtained model is subsequently used by the detection module to evaluate new events or objects or traffic as anomalous or outliers. It is the measurement of deviation that allows classification of events or objects as anomalous or outliers. In particular, the modeling module needs to be adaptive to cope with dynamic scenarios.

3.3 RSSI Data Collection

The main point is to identify anomalous behaviors in the physical layer, thus avoiding the extrapolation of monitoring to higher layers without an individual identification and breaking the confidentiality of customers who are exchanging data. To capture data radio frequencies, a radio probe capable of registering its surroundings is necessary, i.e. **a sniffer**, in this case, a SDR. In addition, a sniffer can be installed on any device connected to a local network, not necessarily the device we want to monitor.

Frequency analysis can be performed on the discrete-time signal by converting the time-domain sequence to an equivalent frequency-domain representation. This can be accomplished with the Fourier Transform of the discrete-time signal. Further processing can produce a spectrogram that shows the power level at given frequencies for a timestamp as shown in Figure 3.5. In the end, we will have a set of frequencies $F = f1, f2, f3, ..., fn$ for a instant of time t, which represents a sweep made by the SDR, resulting in an RSSI array for each frequency, $RSSI(F, t)$ [118] (figure 3.6). The FFT processing carried out to obtain the RSSI values is described by Pedro Martins [119] and the software used for such obtaining is provided by SDR *HackRF* [1].



(a) Ground noise **(b)** Bluetooth Activity

Figure 3.5: Time fragment of frequency spectrum with/without Bluetooth activity outputted by an SDR

[1] https://greatscottgadgets.com/hackrf/

The RSSI is an estimated measure of how good the device can hear, detect and receive signals from any access point. RSSI is one of the most popular and simplest methods for location. The main reason for its popularity is that finding the RSSI requires no additional hardware and can be found on any device utilizing almost any type of wireless communication technology. Since propagating signals are greatly susceptible to noise in the environment, RSSI often leads to inaccurate values that can cause errors in the final result [120].

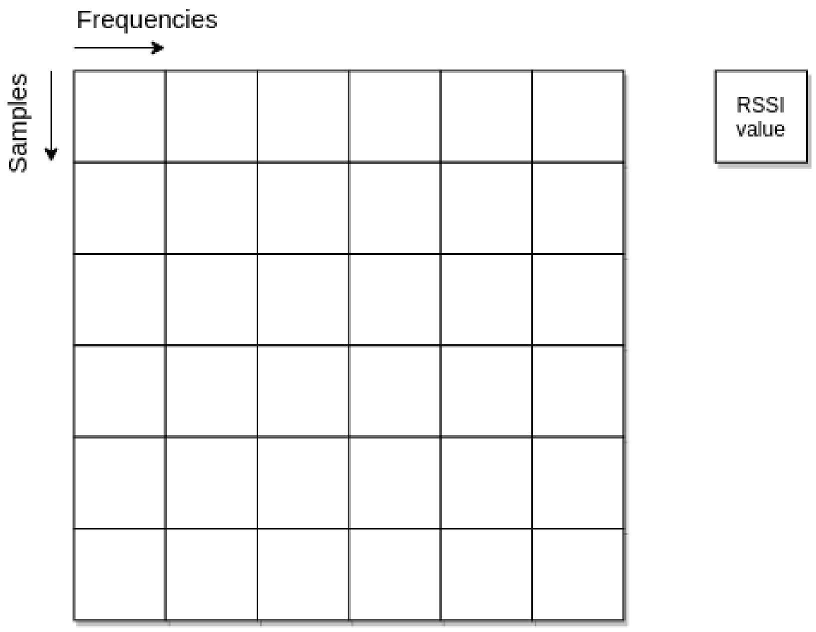
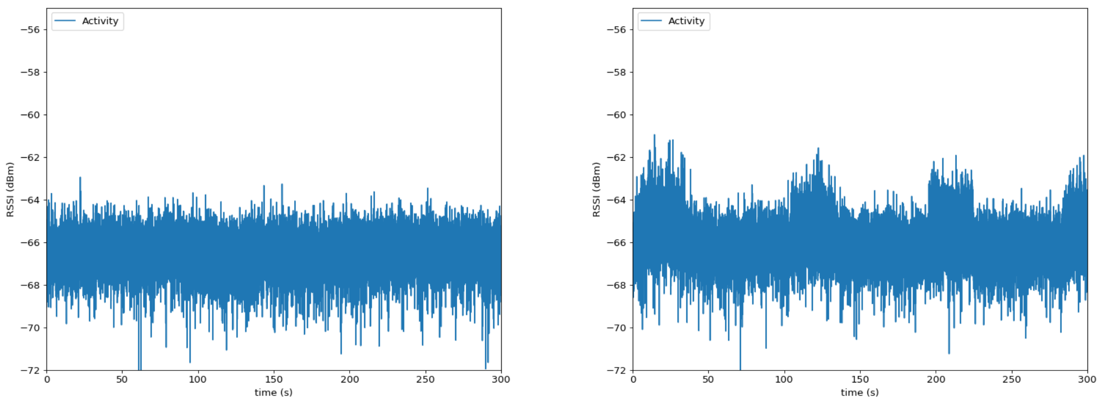


Figure 3.6: Data returned by the SDR

The SDR will output several values, however not all the values are relevant to the problem, depending on the objective, we may only need a representative value. If the goal is to monitoring WiFi channels then the values can be separated into groups(channels) and grouped into an average, resulting in a single RSSI value for each channel. Another scenario is the need to monitor all frequencies and therefore it would be a good idea to group them in small, representative time groups in order to facilitate processing, however losing resolution.

In addition, the SDRs are powered by an amplifier, whose configuration should be carefully done. The point is, assure that is easy to distinguish the ground noise from the network activity, and for that, it's necessary to inspect the outputted signal, while are tested several parameter combinations. The optimal combination who works for a scenario may not be the optimal combination for another scenario. It's nearly impossible to define an unique combination who works for ever scenario. It is expected that an well-configured amplifier facilitates an well-choice threshold, to split the network silence/activity from the ground noise.



(a) Ground noise

(b) Bluetooth Activity

Figure 3.7: Average RSSI values for the entire spectrum for each time point

3.4 Data Preprocessing

Without a context the outputted data from the SDR doesn't have any meaning. It is necessary to have optimal characteristics that represent the data set to be analyzed in order to see an optimal separation of what is clean data in relation to anomalous data, in other words, relevant characteristics.

Working on the direct values given by the SDR is by far the best option given the variations that layer 1 data may have. Collection of RF data can overwhelm even the largest data storage capacities very quickly due to high sampling frequencies. The sampling frequencies can range up to two or even five billion samples per second. Data rates can exceed 200 GB per second and it is expensive to store large samples in real-time [121]. Also, due to physical obstacles and environmental interference, such as the location of transmission that directly influences the propagation of radio waves or urban noise that influences radio wave reception, and even climate change, can have a direct impact on data variation. Depending on the phenomenon, changes in transmission distances may occur, for example, raindrops act as an electromagnetic insulator, reducing the wave's range [122]. In view of the above, performing statistical analysis over absolute power values is not a plausible strategy.

Most anomalies have an underlying periodicity and an amount of data; hence, using statistics of activity and silence moments is favored, as they reveal the behavioral patterns of normal and anomalous traffic. To differentiate activity and silence periods, one must define a RSSI threshold, to separate ground noise from network activity (figure 3.8).

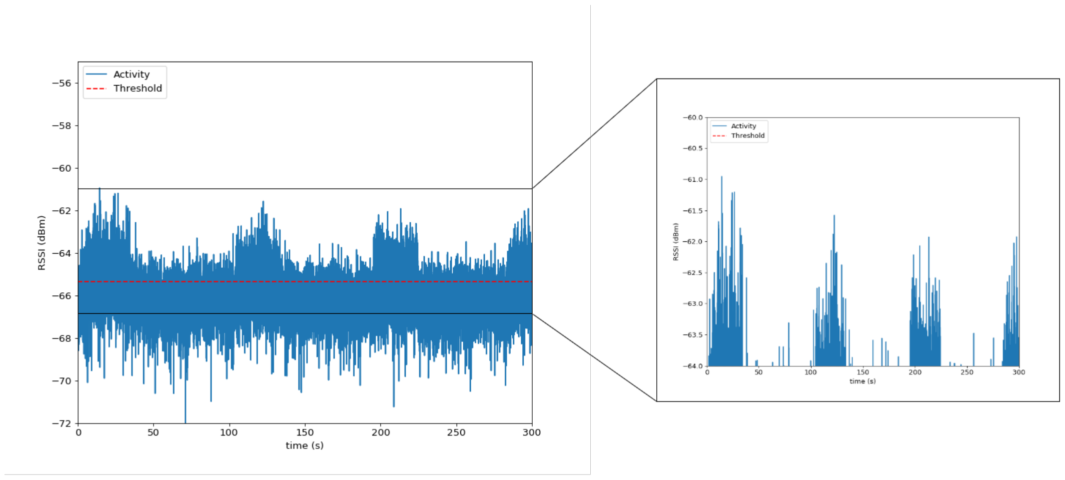


Figure 3.8: Extraction of the relevant periods through a threshold (on the right) of the original capture (on the left)

RSSI values below the threshold will be considered moments of silence and on the contrary, RSSI values above the threshold, are considered an activity. The threshold is given by the equation 3.1, where the w is the max of the RSSI average values, and s is the mean of the activity spikes (above the w). Finally, the threshold is obtained by averaging s,that is, the average of the peaks.

$$\begin{aligned} w &= \max_t \left(\operatorname*{mean}_f \left(RSSI_{f,t} \right) \right) \\ s_t &= \operatorname*{mean}_{f,\text{if } RSSI_{f,t} > 0.95 \cdot w} \left(RSSI_{f,t} \right) \\ thr &= \operatorname*{mean}_t \left(s_t \right) \end{aligned} \tag{3.1}$$

The values above the threshold are transcribed with the value 1, and the values below with the value 0. A period of silence is a block of consecutive samples whose values transcribed are 0, and a period of activity is a block of consecutive samples with values equal to 1.

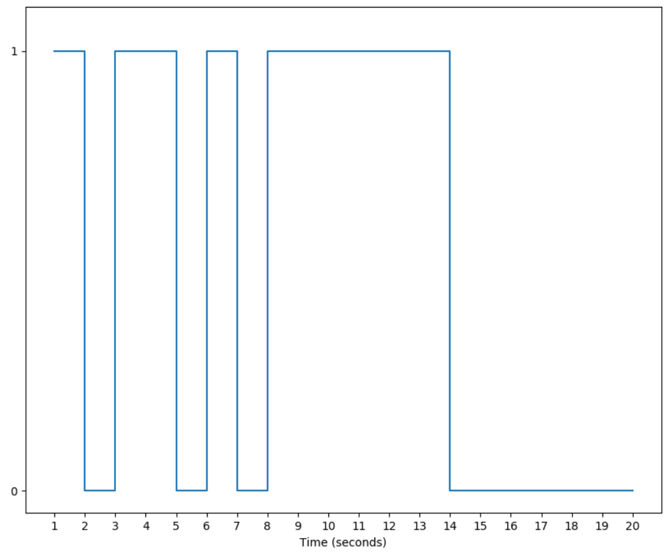


Figure 3.9: Example of post-processing of activity periods.

3.5 Feature Extraction

Even though the activity timelines show the network behavior it's heavy a machine learning model computes all the RSSI values. Large data sets require a lot of computing resources to process. The important thing is that the model retains enough information to learn the behavior patterns of the network.

To perform feature extraction, a period-based approach is used to reduce an activity timelines data set into more manageable groups blocks (**observation windows**), and compute variables into relevant features. The observation window is composed of a set of consecutive time instants called **sub-windows, or sampling window**. The strategy is to split the activity timeline into consecutive observation windows of the size ω, where a window n would start on the sample i and end on the sample $i + \omega$, and the next window $n + 1$ would start on the sample $i + \omega + 1$. However, different windows could have different patterns, making it a hard task for the model to learn.

The solution to this problem, maintaining the similarity of the consecutive windows and increasing the performance, is through the sliding of the windows which will cause a slight overlap. (figure 3.10). A significant advantage of this strategy over a "hard" split is that it usually ends up generating more data windows, which may increase the classification performance. Taking into account the above, and choosing a slide value $\delta < \omega$, the window n would start and end on the same way, although $n + 1$ would start in sample $i + \delta$, and end on the sample $i + \delta + \omega$.

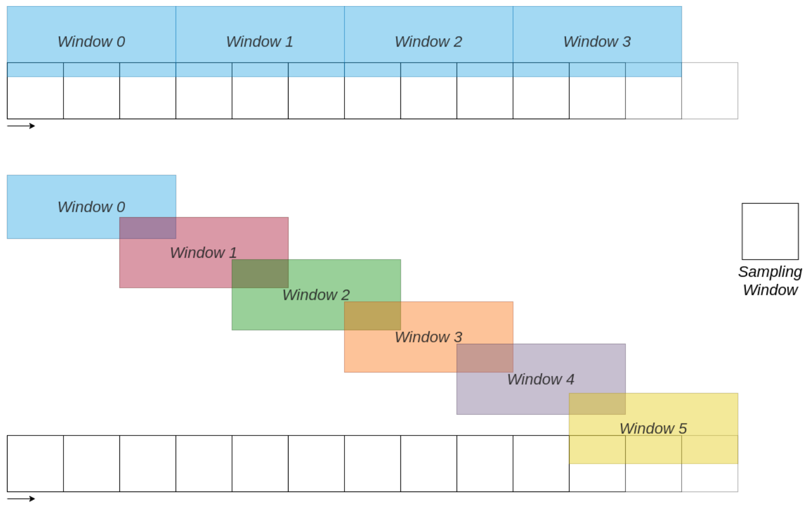


Figure 3.10: Comparison of how the timeline samples are distributed across data windows. On the top, a "hard" separation. On the bottom, the sliding-based approach.

In this light, the proposed method permits performing a statistical analysis over the number of frequencies and the periods of silence and activity allowing differentiating behaviors through distinct periodicities. Feature extraction over multiple window sizes is a simple approach, since they provide insights in different periods. Nonetheless, the dataset must be split according to the most extensive window size and the defined window slide. For every observation window, the statistical measurements are computed (figure 3.11) .

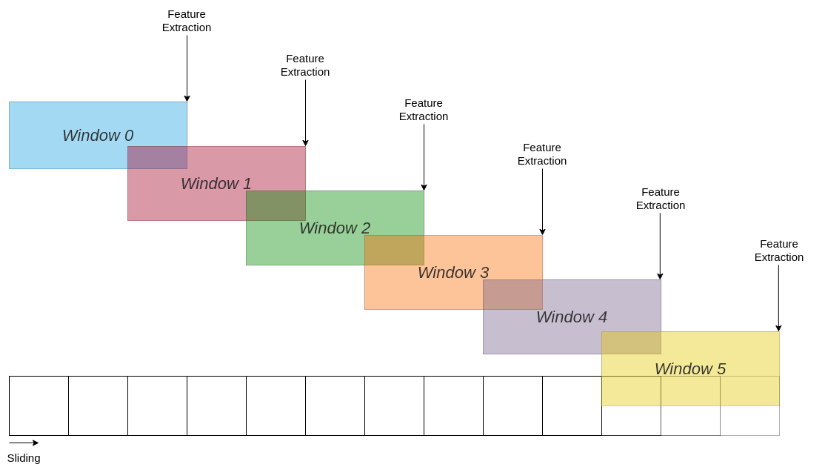
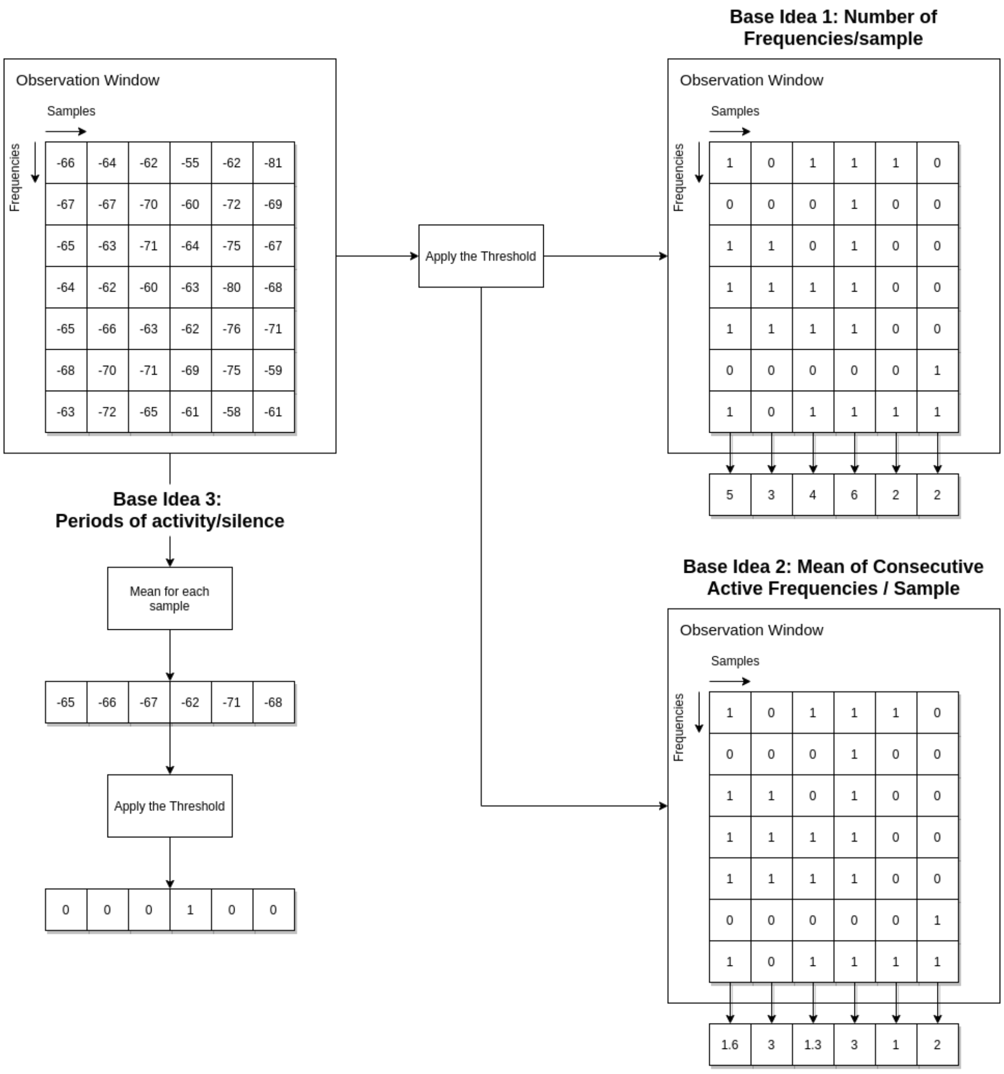


Figure 3.11: Diagram of the moments when the features are extracted.

The resources extracted from each observation window have three basic information sources based on the behavior of communication technologies and the behavior of the network, the **number of active frequencies**, **consecutive active frequencies**, and **activity/silence periods** (figure 3.12). WiFi uses a restricted spectrum range, so when WiFi is used, there are fewer frequencies to "activate" compared to Bluetooth that uses the full spectrum, and therefore a greater number of consecutive active frequencies. Based on this assumption, the first two information sources for feature extraction were created, however these statements could suffer from external influence if we have more than one Access Point (AP). The third information source assumes that the silence of a network, that is, the time that there are no active frequencies is much less in the normal behavior of a network, since nowadays there is almost uninterrupted use of the Internet, directly or indirectly.



Base Idea 1: Number of Frequencies/sample
Base Idea 2: Mean of Consecutive Active Frequencies / Sample
Base Idea 3: Periods of activity/silence
Observation Window
Samples
Frequencies
Apply the Threshold
Mean for each sample

Figure 3.12: The three basic forms that will be worked on in each observation window. The 1st form is based on the "activated" frequencies and the 2nd form is based on the average of consecutive active frequencies, and the third is based on the periods of activity and silence .

As showcased in figure 3.13, there are three types of features for each base information source, **active/inactive frequency features**, **dispersion features**, and **activity/silence features**. However, each of them is computed in three different ways, either based on each base information source.

Mean, median, standard deviation, 75º, 90º, 95º and 99º percentiles, minimum and maximum are values referring to periods of activity and silence or to active and inactive frequencies. In addition, both communication technologies present different ways of using the spectrum e.g. Bluetooth is used for data transmission through short range UHF radio waves between 2.4 and 2.485GHz, occupying very similar frequencies with WiFi, that uses one channel at a time. In principle, different wireless communication channels has different behaviors on the range.

Therefore, an interesting way to differentiate them, as was said before, is to see how the active frequencies are dispersed. They have different frequency range sizes, so Bluetooth will have a higher **difference between the first and last positions** in the list of frequencies.

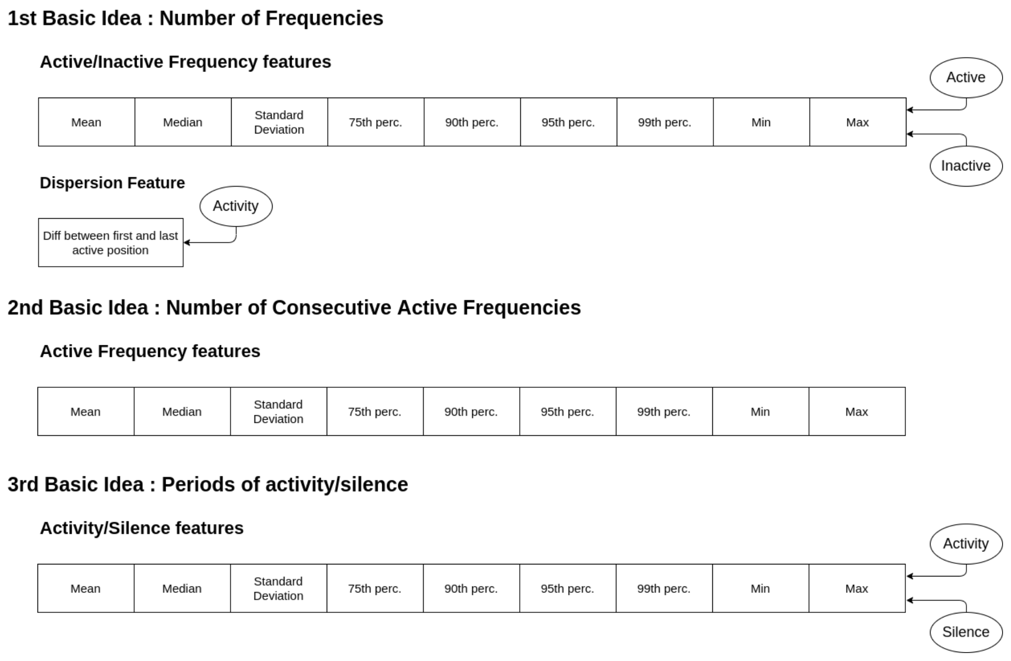


Figure 3.13: Diagram of features extracted for every sub-window size.

3.6 Feature Dimensionality

As mentioned in 2.5.2, in machine learning to obtain a more accurate result we trend to add as many features as possible as first. However, after a certain point, the performance of the model decrease with the increasing of features, known as "*The Curse of Dimensionality*". Curse of dimensionality describes the phenomenon where the feature space becomes increasingly sparse for an increasing number of dimensions of a fixed-size training dataset. On the other hand, if the training data is not enough, there is a risk to produce a model that could be very good at predicting the target class on the training dataset but fail miserably when faced with new data.

So to overcome the curse of dimensionality and avoid the overfitting it is a good practice to perform some sort of **dimensionality reduction**, in fact, the learning process will be significantly faster with a smaller number of features and yields a more general concept. This helps in getting a better insight into the underlying concept of a real-world classification problem [123].

In the previous section, 3.5, we considered that there are n observation windows in each data window, and for each observation window are extracted m features, having a total of $n \times m$ features on each data window. It is probable that most of the features are correlated or not have statistical relevance to be considered. For this situation, it can be beneficial to use only a subset of $k < m$ of those features [124].

In section 2.5.2, two approaches are presented to reduce the total amount of resources used to train the models. The first approach PCA, which maps a higher dimension set of features to a smaller one without losing the information of the original dataset. The second strategy is through the tree-based classifiers, using only those given by a series of models that output the relevance of each input feature, to train the classifiers. The idea behind using multiple models is to ensemble different perspectives into a reliable global selection methodology. Both strategies should be tested to assess which one performs better.

Since the main objective of this dissertation is to develop an anomaly detection system, anomaly samples should not be used during model training for the classifier to learn only to classify clean data and differentiate them from anomalies. The idea is not to classify clean data from anomalous data, but to distinguish clean data from anomalous data. However, the data set used to model feature selection will have both clean and anomalous data, preferably of different types to avoid the tendency to select features for a given target, however it's not possible obtain every possible anomaly. The use of this data set aims to verify which resources are potentially most useful for differentiation.

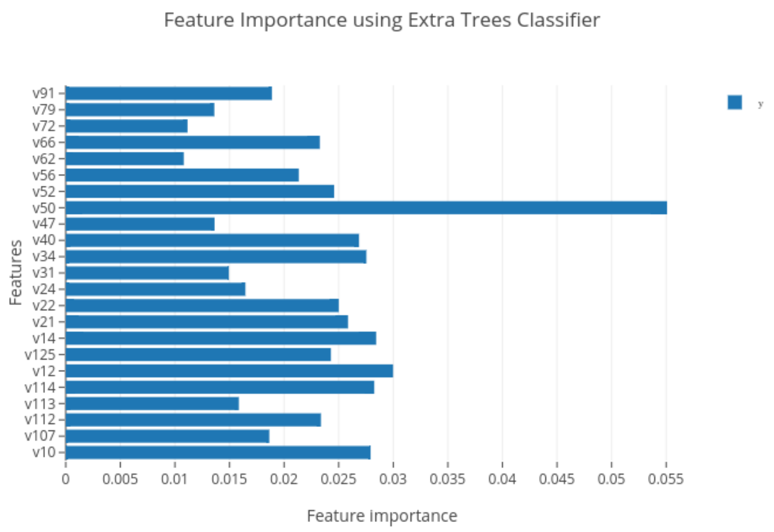


Figure 3.14: Feature importance using ETC (retrieved from Plotly website [2])

The majority of embedded feature selection models return a normalized importance value. However, these values do not translate very well across different data sets (e.g., a given feature may have different relevance in different data sets). To solve this, is proposed a classification system. The idea comes from the problem already presented in section 2.5.2, *Feature Selection*,

[2]https://chart-studio.plotly.com/

on the stability of the feature selection algorithms. *Feng Yang* and *K.Z. Mao* [84] proposed a fusion of multiple criteria based on scoring, where "*each base criterion first produces a scoring vector containing scores from all resources, a scoring combination algorithm is then employed to aggregate the multiple scoring vectors score in a consensus score vector, and a resource ranking procedure is finally carried out to rank resources based on their consensus scores*" as illustrated in figure 3.15.

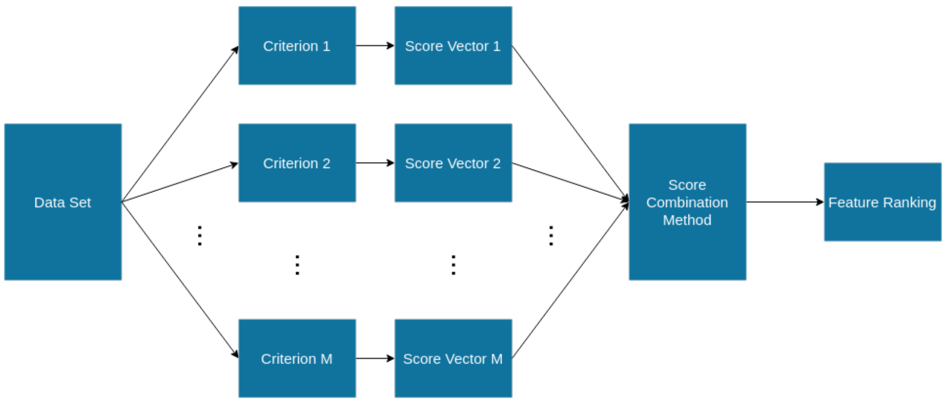
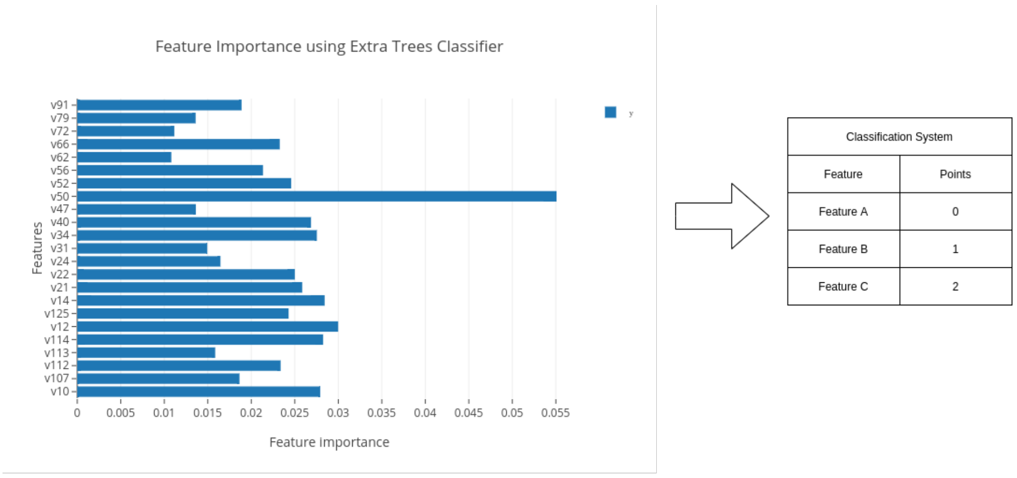


Figure 3.15: Score-based multicriterion fusion.

For N different data sets, each characteristic receives points, which corresponds to the position of the characteristic of how much more relevant it is to the dataset. In the end, the K features with the fewer points are selected as the most relevant. This process is repeated n times, to be assured that the features chosen are the best ones (figure 3.16).



Classification System	
Feature	Points
Feature A	0
Feature B	1
Feature C	2

Figure 3.16: Points-based Classification System to find the best features to model our problem

The K value will be the minimum number of resources to achieve good performance, but with reasonably fast learning times. The objective, will be verify which K optimizes it's classification *F1-Score*.

3.7 Model training and Evaluation

Once the original data is reduced in size, the data is prepared to be processed by machine learning algorithms. First, the clean data will be divided into training α and test 1-α sets. Even we are only considering only OCC models, the anomaly data will be divided into training and testing sets for Cross-validation (CV) purposes.

Each model is trained n times, using different training and testing sets, making it possible to reduce the performance metrics variance and return a confidence interval of those results.

For each training iteration, after splitting the data sets (one containing only "normal" behavior and other containing only anomalies) into training and testing sets, classifiers are training with multiple combinations of hyper parameters. The goal is to find the set $\mu_{group} \in S$, i.e. find the best arguments, at which the function output *F1-Score* is the best as possible, where S is a group of sets of hyper parameters, and i the number of splits, that defines a model C maximizing the average *F1-Score (F1)* on multiple sets of CV.

$$\mu_{group} = \underset{\mu \in S}{\operatorname{argmax}} \frac{1}{i} \sum_{(x,y) \in \mathrm{CV}} F1(C, \mu, x, y)$$

For each combination, the "normal" training set is divided into training and CV sets using K-Fold Cross-Validation splitting (figure 3.17), which randomly divides a data set into k disjoint folds with approximately equal size, and each fold is in turn used to test the model induced from the other $k-1$ folds by a classification algorithm. The performance of the classification algorithm is evaluated by the average of the k accuracies resulting from k-fold cross validation, and hence the level of averaging is assumed to be at fold. All folds are assumed to contain the same number of instances except explicitly specified [125].

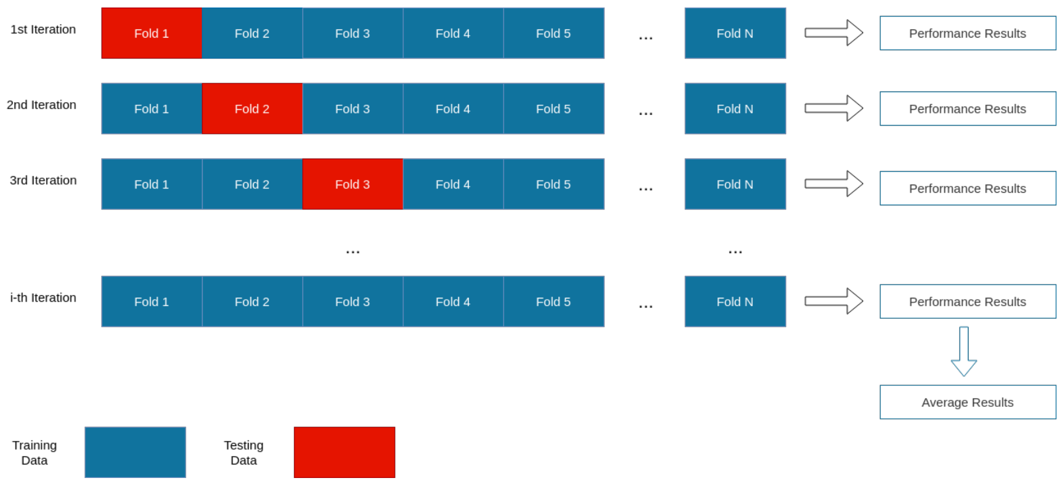


Figure 3.17: Illustration of the cross-validation division

To assess the performance of each combination of hyper parameters, both anomalous and clean data will be used, ideally split the data into 70/30 or 80/20. The splitting are exemplified in equations 3.2 and illustrated in figure 3.18 and is repeated n times. With this approach there is a possibility of high bias if we have limited data, because we would miss some information about the data which we have not used for training. If our data is huge and our test sample and train sample has the same distribution then this approach is acceptable.

$$\begin{aligned} cv_{trainSize} &= \mu \cdot cleanSet_{size} \\ cv_{test\_CleanSize} &= (1-\mu) \cdot cleanSet_{size} \\ cv_{test\_OutlierSize} &= (1-\mu) \cdot anomalySet_{size} \end{aligned} \tag{3.2}$$

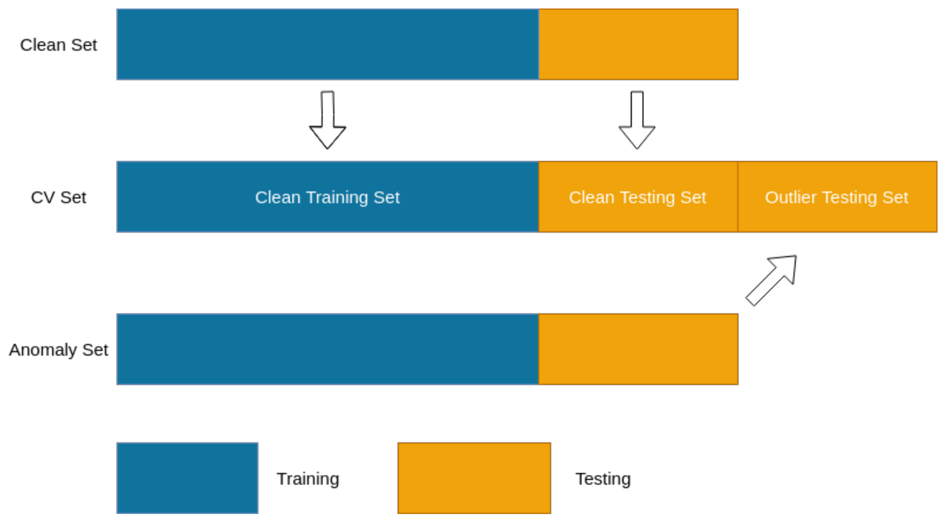


Figure 3.18: Splitting of the training data sets into training and cross-validation sets

Beside feature selection and dimensionality reduction, an important technique to improve significantly the performance of some models is **feature scaling**, to handle highly varying magnitudes, values, or units of the features. Some machine learning algorithms are sensitive to feature scaling while others are virtually invariant to it, p.e. machine learning algorithms like KNN, K-means, and SVM, among others distance-based algorithms, are most affected by the range of features, because they are using distances between data points to determine their similarity. For multiple features with different scales, higher weightage is given to features with higher magnitude and this will impact the performance of the machine learning algorithm and we do not want the algorithm to be biassed towards one feature. We scale the data before employing distance-based algorithms to all features that contribute equally to the result. On the other hand, tree-based algorithms are fairly insensitive to the scale of the features. A decision tree is only splitting a node based on a single feature, a feature that increases the homogeneity of the node. This split on a feature is not influenced by other features. This is what makes them invariant to the scale of the features.

There are two commonly used feature scaling techniques in machine learning. **Standardization** can be helpful in cases where the data follows a Gaussian distribution, however, this is not necessarily true. Unlike Normalization, standardization does not have a bounding range, so even with the presence of outliers, they will not be affected by standardization, but in the other hand, **Normalization** can be helpful in algorithms that not assume any distribution of the data like KNN or Neural Networks [126].

Since the main objective of this dissertation is to fit the normal behavior of the network in the best possible way, it is crucial not to contain any anomaly data, being, as previously mentioned, the one-class classifiers the optimal solution for this dissertation. With the right set of resources, it will be possible to detect invisible phenomena and, consequently, trigger alarms for possible outliers.

The presented models in section 2.5, in the optimal scenario, should be able to detect any anomalies, seen and unseen phenomena, and not trigger alarms by mistake, i.e., is expected **small values for false positive and false negatives**. Each classifier will be classified and comparing according to its *F1-Score*. Consequently, both precision and recall must be computed according to their number of outliers True Positives (TP), False Positives (FP), and False Negatives (FN), as showcased in equation 3.3.

$$\begin{aligned} precision &= \frac{TP}{TP + FP} \\ recall &= \frac{TP}{TP + FN} \\ F1 &= \frac{2 \cdot precision \cdot recall}{precision + recall} \end{aligned} \tag{3.3}$$

As stated before, the model will be trained and tested with different data sets resulting in an average result and a confidence interval according to the equation 3.4, where $\overline{x}$ and σ is the mean and standard deviation, respectively, of a set of performance results, n is the size of the set and Z, a tabulated value that corresponds to the probability that an interval for a statistical sample actually includes the parameter population.

$$IC = \overline{x} \pm Z_C \cdot \frac{\sigma}{\sqrt{n}} \tag{3.4}$$

3.8 Summary

The mechanisms presented throughout this chapter aim at filling a gap regarding physical layer anomaly detection strategies, since, up until this moment, we were not able to find any work that used RSSI data for pattern and behavioral analysis, rather than location-based outlier detection. With this in mind, the main aspects of the proposed mechanisms are:

- These are mechanisms focused on privacy because at no time is the information contained in the network's data flow revealed, making the identification of individual behavior a challenge for a network with multiple devices sharing the same spectrum.
- It focuses on the analysis of the network behavior and wireless communication behavior, that is, the modeling of periods of activity/silence and active/inactive frequencies extracted from the time series RSSI and the arrangement of wireless communication channels in the spectrum.
- It can be adapted to other radio technologies with relative ease, not only WiFi and Bluetooth, mainly because it uses SDR for capturing physical-layer data.

- It is not necessary to feed the models with every type of anomalous behavior imaginable, only with clean data. However, these mechanisms cannot work by themselves, an extra level of verification, manual or automatic, is required to complement the mechanism, to verify the appearance of false negatives.
- It focuses on optimizing the model detection performance based on metrics that work even on unbalanced data sets, such as the *F1-Score*.

The proposed mechanisms will be further explored in the next chapter, where they will be employed to perform outlier analysis in a home automation set scenario.

CHAPTER 4

Methodologies Evaluation

In order to validate the anomaly detection methodology presented in the previous chapter (chapter 3), home automation scenario were tested, in order to verify whether the proposed methodology can identify periods of anomalous behavior across the spectrum at the 2.4GHz frequency.

In addition, we will do a complete analysis of the recorded data; this will include the entire pipeline, from registering the data captured on the network to obtaining the relevance of the extracted resources. Furthermore, performance metrics are presented in terms of detection *Accuracy*, *Precision*, *Recall*, and *F-score*. Finally, overall discussion on the result and a summary concludes the chapter.

4.1 IoT Home Scenario

This scenario serves only as a proof of concept. It reflects a home automation scenario, represented by a network composed of a laptop (Legitimate PC) with access to the Internet through an AP, generating normal traffic, and a mobile phone that has been compromised that exchange files with a laptop (Pirate PC) through a Bluetooth connection, generating anomalous traffic. The normal behavior of the network is given by frequent interactions of the Legitimate PC, running Linux, with the AP, through the uninterrupted playback of *Youtube* videos or with no activity in the network, while the anomalous behavior is given by the periodic file transfer between the Pirate PC and the mobile phone.

Initially, datasets will be created with isolated events, namely a dataset exclusively with Wi-Fi traffic or ground noise, i.e., the interaction of the laptop with the AP will represent a clean dataset and a dataset containing exclusively anomalous behavior during a fixed period will represent a dirty dataset.

Posteriorly was blended the clean behavior with the anomalous behavior by keeping the interactions between the compromised personal computer and the AP in the background at the same time the mobile phone change files with the personal computer. (figure 4.1).

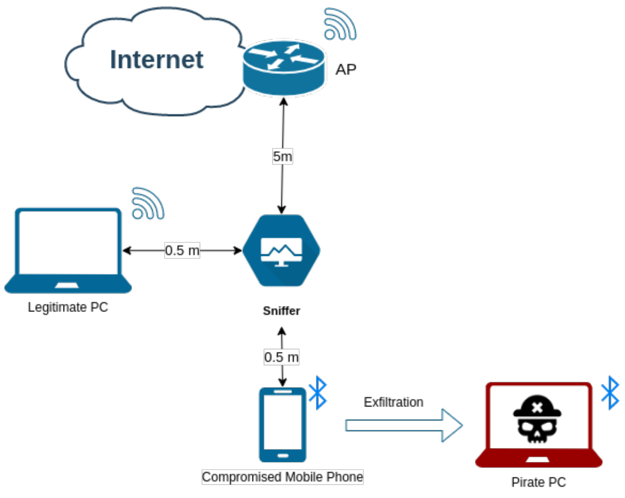


Figure 4.1: Home Scenario Network Diagram

In order to simulate data exfiltration from the compromised device, it's simulated a transfer of files from the cell phone to attackers computer, with the following distribution statistics:

- The transferred file has 5MB;
- A file transfer which occurred every 120 and 300 seconds, hereafter referred as *scheduled 120s* and *scheduled 300s*, respectively;

In this simulation, the sniffer that we used to perform the signal captures is the *HackRF One* [1], an open-source hardware SDR, capable of transmission or reception of radio signals from 1MHz to 6GHz, that can be used as USB peripheral, performing live FFT computations over the outputed data. The tests were performed over the 2.4GHz band, more specifically between 2401 MHz and 2483 MHz , since Bluetooth uses the 2.4 GHz ISM spectrum band, and the Wi-Fi channel was set to channel 8. Channel 8 is "free", i.e., no one device is working on that channel. Another aspect to consider is the sample rate, which was set to 20MHz, so that it could cover the maximum range of frequencies. This was also the maximum allowed value by the *HackRF One*.

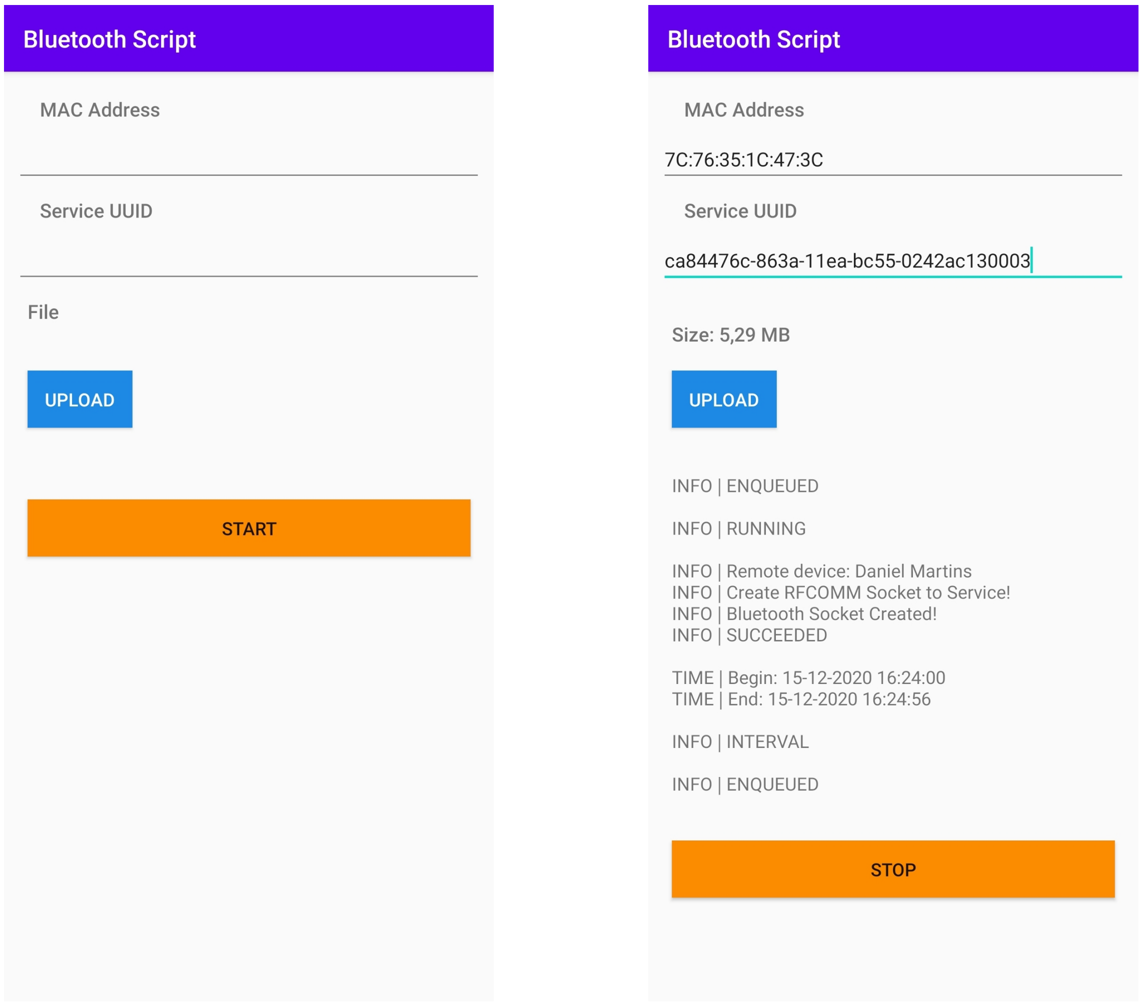
When calibrating the *HackRF One*, the goal is to separate the ground noise from activity in the spectrum. In the conducted tests, its amplifier Low-Noise Amplifier (LNA) gain was set to 32dB, and the Variable-Gain Amplifier (VGA) gain to 16dB. The LNA corresponds to the main gain, and the VGA to the ground-noise. Several combinations were made and, after visualizing the graphical representations of the output signal, these values proved to be the best for the proof-of-concept scenario mentioned above.

The place where the sniffer is placed is a very important aspect, which should not be too far from the AP, as it will affect the signal captured and make noise an obstacle to the proof-of-concept study. Thus, the sniffer was placed 0.5 meters from the nodes and 5 meters from the AP, in-between them.

The simulation of an attack was through the development of an android application that periodically sends a file with a specific size to the pirate PC. The automated periodic sending

[1]https://greatscottgadgets.com/hackrf/

was developed through a Python server which provides a Bluetooth service with a specific identifier to which the phone will connect. The android application through the MAC address and the Bluetooth service identifier is able to establish a connection with the service. As soon as the connection is established, the transfer begins.



MAC Address: 7C:76:35:1C:47:3C
Service UUID: ca84476c-863a-11ea-bc55-0242ac130003
Size: 5,29 MB

```
INFO | ENQUEUED

INFO | RUNNING

INFO | Remote device: Daniel Martins
INFO | Create RFCOMM Socket to Service!
INFO | Bluetooth Socket Created!
INFO | SUCCEEDED

TIME | Begin: 15-12-2020 16:24:00
TIME | End: 15-12-2020 16:24:56

INFO | INTERVAL

INFO | ENQUEUED
```

Figure 4.2: Android application for periodic file uploads

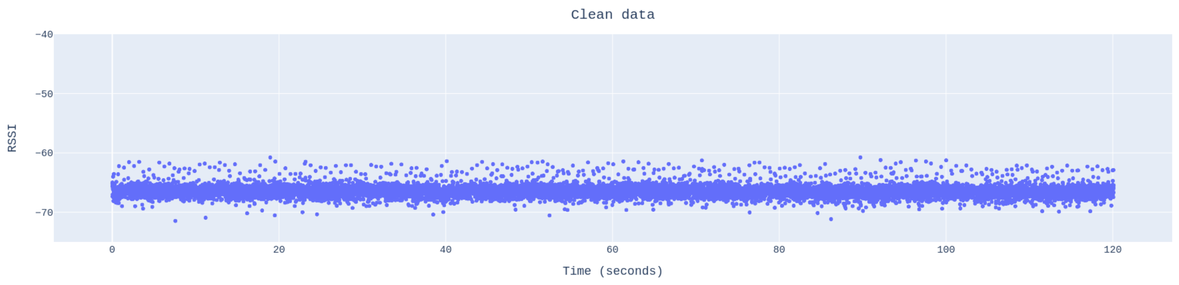
4.1.1 Data Exploration

All data sets referring to the home automation scenario have 1 hour of records. However for visualization purposes, graphs of a few minutes will be shown, unless otherwise indicated.

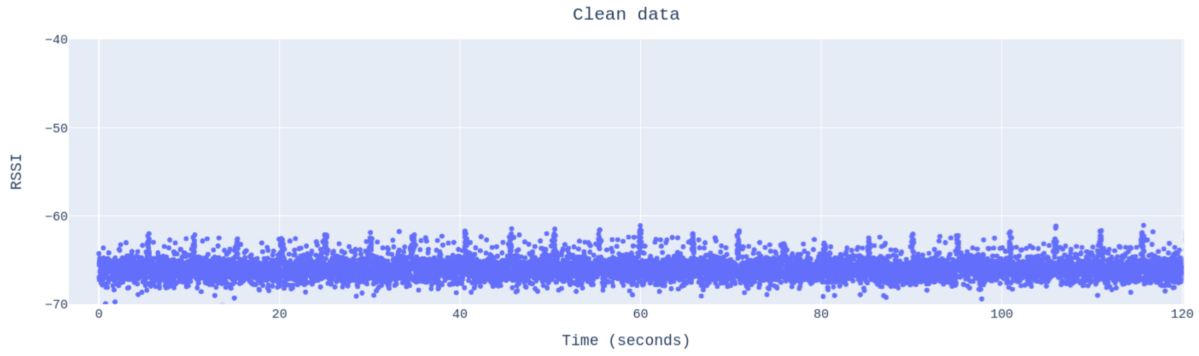
The FFT processing of the IQ data outputted by the *HackRF One* resulted in a file containing several samples, each one containing RSSI values of every frequency of interest, with approximately 81 samples per second. In sum, we have 83 RSSI values, each corresponding to a frequency. The values are stored in a matrix, each cell corresponding to an RSSI value from a frequency in a specific time instant.

To understand and visually analyze the behavior of the network over time and validate the data to later train and test the classifiers, these 83 values can be calculated to obtain an average RSSI across the spectrum for each instant of time.

As illustrated in figures 4.3 and 4.4, the outliers have completely different behavior than the typical traffic characteristics of the network. The first plot, 4.3a, mainly shows the 2.4GHz range without a single data exchange, only ground noise and control and management packages, which maintain the credibility of the scenario; the second 4.3b is playing a *Youtube* video. Youtube's behavior is described by activity spikes, periodic. The *Youtube* behavior transmission is done by packets of a few seconds that are in the constant buffer. As the video plays, the buffer is empty and YouTube reloads it. This allows you to change the quality. That is, if we have 40 seconds of a movie at 360p and we want to improve the images, you can switch to 720p and *Youtube* will send packages with a different quality. A few moments later, the video - still playing - will have improvements in resolution and will not show any pauses. Therefore, only from X to X time is the buffer filled again. That's the reason why YouTube's behavior shows spikes in activity. In contrast, both third 4.4a and fourth 4.4b plot shows two different outliers, the first with short data bursts every 120 seconds, and the second 300 seconds.

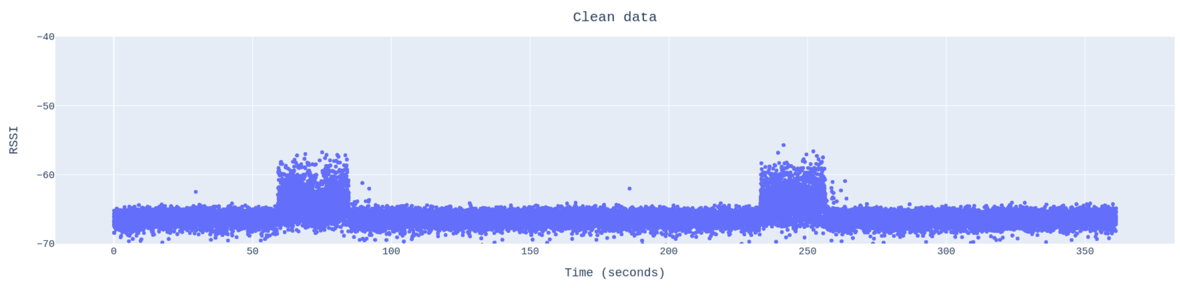


(a) Network without any activity

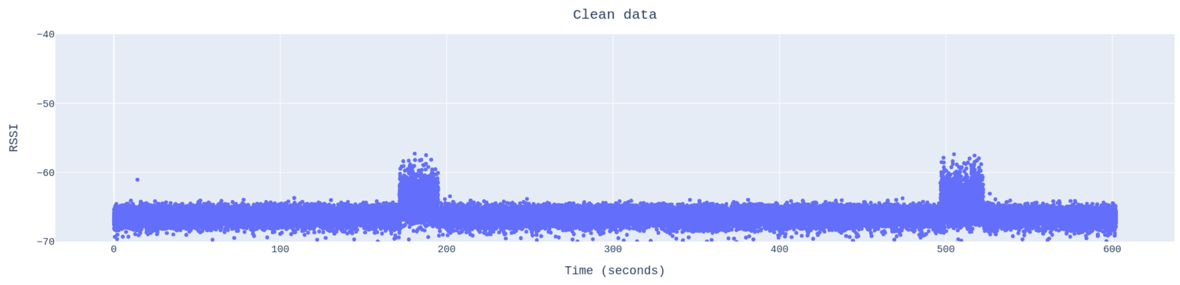


(b) Network with a single activity

Figure 4.3: RSSI power values over time in two different data sets exclusively containing clean traffic.



(a) Anomaly Behavior (120s scheduled outlier)



(b) Anomaly Behavior (300s scheduled outlier)

Figure 4.4: RSSI power values over time in two different data sets exclusively with anomalous behavior.

Figure 4.5 now show the spectrum behavior when the compromised node is running in the background, masked by the usual behavior of the network. Besides the typical interactions with *Youtube*, the anomalous behavior is also present which overlap the non-anomaly behavior.

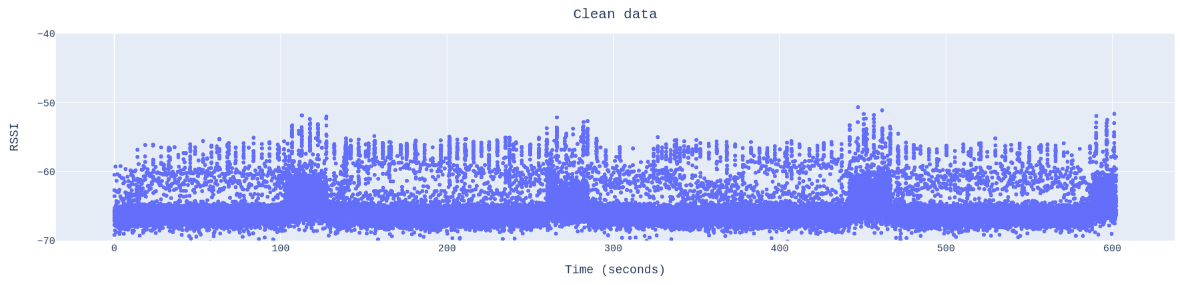


Figure 4.5: RSSI power values over time with anomalous traffic masked behind typical network traffic of one node, Linux laptop streaming YouTube videos.

The reproduction of *Youtube* videos almost completely masks the anomalous behaviors, besides the *Youtube* videos is also periodic, much like the anomalies, which makes it harder for the models to identify these patterns.

Threshold

The next stage is to extract the relevant information in a way to get the three basic information sources from the data series data. As mentioned in the previous chapter, it's necessary to establish a threshold to separate activity from silence. Equation 3.1 presented in the previous chapter makes it possible to differentiate activity from silence.

The data is split into sliding windows of 10 minutes with 5 seconds sliding distance, with a sampling window of 1 second. Since we don't know the frequency of anomalies, a 10 minute observation window is proposed to have a greater field of view about a possible anomaly, with an offset of 5 seconds, which despite generating more data, it is extremely important to extract information and make a decision as soon as possible to prevent data exfiltration. A 1 second sampling window is proposed in order to reduce recorded information. Without applying a sampling window, 81 samples per second would be recorded. Registering each sample is disadvantageous for later resource extraction. It is not possible to make a greater reduction, as we would be losing information and possibly losing anomalies.

As showcased in section 3.5, activity/silence features, dispersion features and trading features over any of the windows are extracted from the time-series data.

The relevance of the features

Not all these features are equally relevant for detecting outliers. In section 3.5 some assumptions were made about how clean and anomalous behaviors can be differentiated. As mentioned so far, it has been put forward that different communication technologies will reveal a total number of different active frequencies. Wi-Fi makes use of a restricted number of active frequencies, while Bluetooth makes use of almost the entire 2.4GHz range. Figure 4.6 shows that the use of Bluetooth has an average of active frequencies higher than Wi-Fi. Contrary to what was supposed, the number of active frequencies is far from the size of the used range.

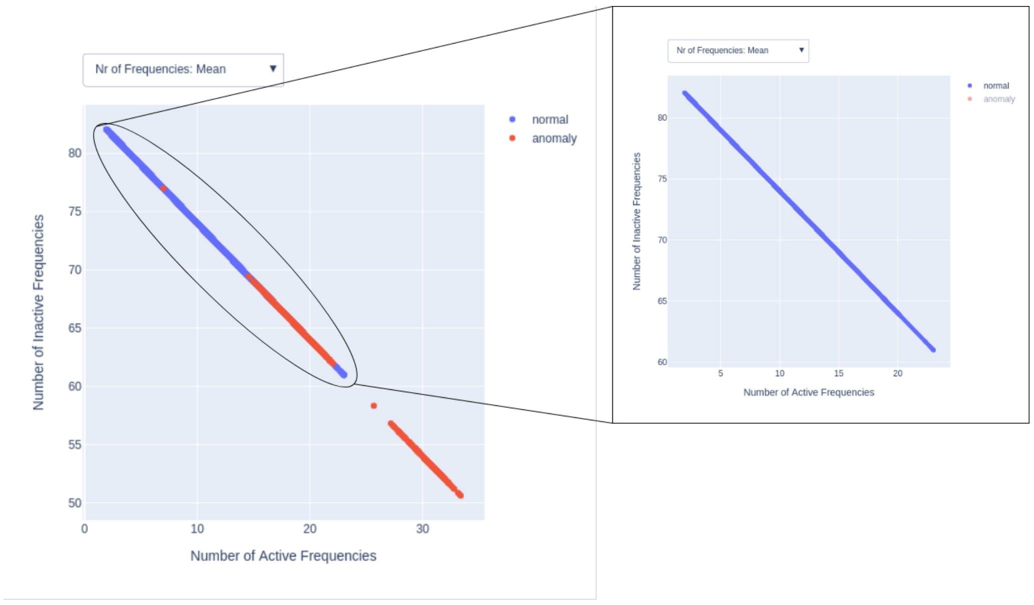
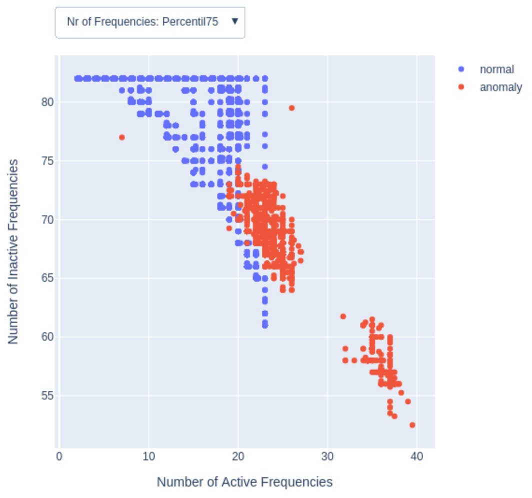


Figure 4.6: Comparison of the number of active and inactive frequencies in clean and outlier behaviors.

A *Wi-Fi* channel uses approximately 22 frequencies, and figure 4.7 shows an average of active frequencies between 2 and 23, reaching maximum values between 8 and 27, which reveals the presence of noise in the dataset. The extra used frequencies can be explained by radio noise and random Wi-Fi activities from active devices in the vicinity. It is noteworthy that the figures there are anomalous data overlapping the normal ones, revealing that there are periods in which the number of active and inactive frequencies is identical in both behaviors, which can create an obstacle to differentiation, and possibly the generation of false negative.



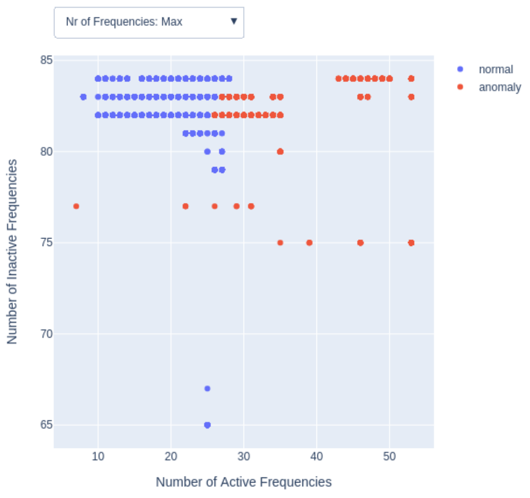


Figure 4.7: Percentile 75 and maximum number of frequencies in clean and outlier behaviors.

A differentiating factor that goes according to the assumption made, is the DFLP. The difference between the first and last position of the active frequencies is notorious. It is not possible to conclude whether this same metric would work to distinguish if the network had more APs. In our scenario, the clean behavior presents a value interval between 2 and 46, and the anomalous behavior present an average between 34 and 71 positions (figure 4.8). As mentioned before, a Wi-Fi channel has a total of 22 frequencies, however the average difference between the first and the last active position is much higher than this limit. Possibly the presence of background noise explains the data presented between the third quantile and the maximum value.

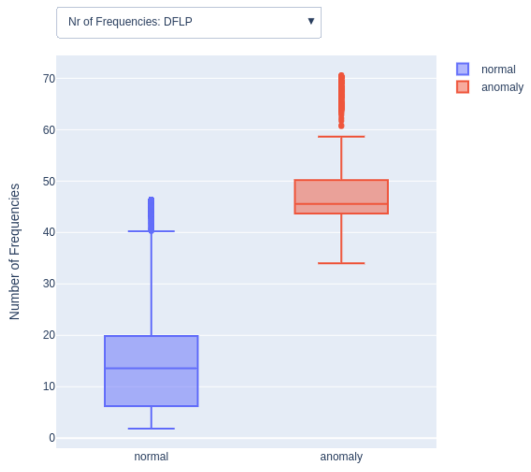
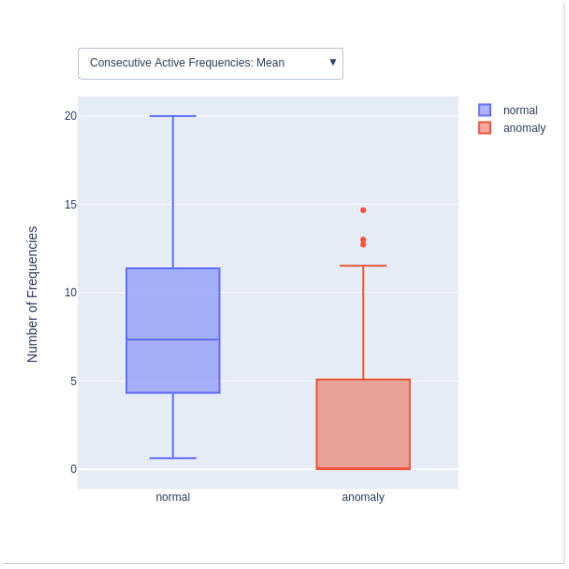
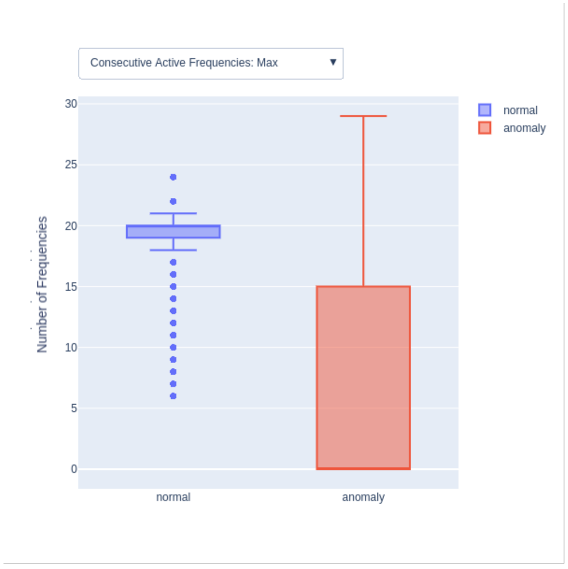


Figure 4.8: Mean difference between the first and the last active frequency in clean and outlier behaviors.

The second assumption was based on the number of consecutive active frequencies. Since Wi-Fi uses fewer frequencies, then the number of consecutive frequencies will be greater than Bluetooth. This assumption holds even for an increase in the number of APs. Through figure 4.9a, it can be concluded that the assumption made is true. The number of consecutive frequencies is different in both behaviors. Figure 4.9b shows that maximum values support the assumption made. The normal behavior remains among the 20 consecutive active frequencies while the anomalous behavior presents more dispersed values, presenting a higher density between the median and the 3rd quantile, but can reach a maximum of 30 consecutive active frequencies.



(a) Average number of consecutive frequencies.



(b) Maximum number of consecutive frequencies.

Figure 4.9

Another important factor for the distinction between both behaviors is the times of activity and silence. The third assumption is based on the idea that the iterations generated by the incessant reproduction of *Youtube*, will cause a longer time of activity and a shorter time of silence. In contrast, the anomalous behavior generated by the periodic sending of a file will cause shorter activity times, but longer periods of silence.

Analyzing this case, figures 4.10 and 4.11 reveal what was expected, long periods of silence and short periods of activity on the part of anomaly behavior. Oh the other hand, the normal behavior show longer activity times and shorter silence times. One noticeable thing on these plots is that there is a correlation between activity and silence periods; considering that window sizes are fixed, more extended activity periods results in shorter silent periods, and vice-versa.

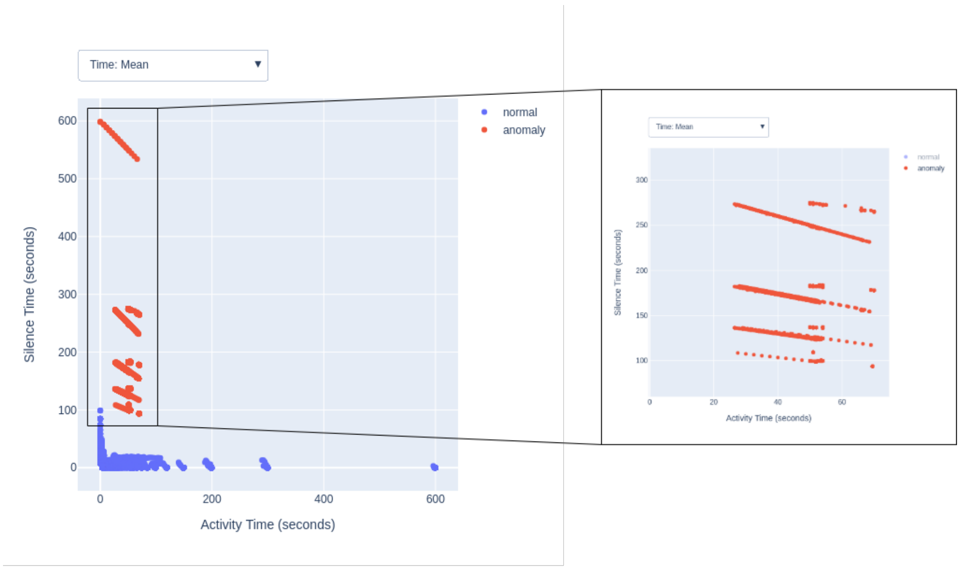


Figure 4.10: Comparison of the mean silence and activity periods on clean and outlier behaviors (anomaly with a with a periodicity of 300 seconds).

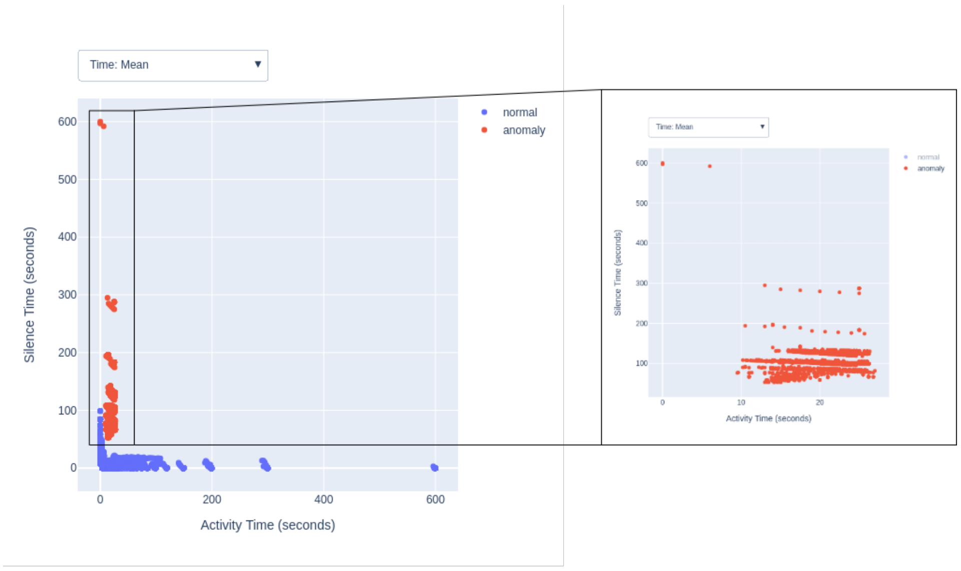


Figure 4.11: Comparison of the mean silence and activity periods on clean and outlier behaviors (anomaly with a with a periodicity of 120 seconds).

The data demonstrate also a poorly controlled environment. In a fixed window, with an anomaly with an activity time between 50 and 60 seconds - time taken to transfer a 5MB file - and an interval between anomalies of 300 seconds, only two cases can happen, or you have an anomaly or two in a single window. Theoretically, the activity time in a 10 minute window can vary between an average minimum of 27.5 seconds, corresponding to the presence of a complete anomaly plus 5 seconds of a new anomaly - an observation window slides 5 seconds - and a average maximum of 60 seconds, corresponding to the presence of one or two complete anomalies. Looking closely for the figure 4.10, it is possible to notice that the activity time is longer than stipulated by theoretical calculations. Having said that, there are two factors that may have influenced this result. The software developed for the periodic sending of files over a Bluetooth connection, at specific times, may have failed and generated delays, or there is the possibility of noise being present, generating unexpected activities. For example, in figures 4.10 and 4.11 there are data with 600 seconds of silence and 0 seconds of activity proving without a doubt the software delay, because the software starts to transfer the 5MB file as soon as the capture starts, and therefore it will generate a dataset with anomalies every 300 seconds without stopping, therefore it is "impossible" to generate observation windows without any activity.

In general, the average times of both anomalies are similar, mainly in relation to the time of silence. If the transferred file were smaller, that is, the information exfiltrated was smaller, it would still be possible to differentiate between the two behaviors, being an enhancing characteristic for the differentiation of both.

The anomalous behavior present in all indicators analyzed so far presents data similar to the normal behavior. This behavior can be explained by the way the features of each observation window are extracted. First, each point on the graph is an observation window. In a 10 minute observation window, if there is at least 1 second anomalous, the entire window is considered anomalous, and therefore the statistically similar points in the graphs so far revised, can be explained by this method of extracting features.

To conclude, all the assumptions proved to be true, however not all proved to be robust. The times of activity and silence were more promising, which could be a possibility if this methodology is used in more than one scenario, namely with the increase of APs and Bluetooth connections. The basic idea in relation to the number of frequencies used proved to suffer from external influences, with the possibility of generating a large percentage of false negatives.

4.1.2 Feature Dimensionality

There are a variety of ensemble models that provide feature-importance information, such as Extra Trees, Decision Forests or Random Forest classifiers that complement the feature classification proposed in section 3.6 and given the number of features, we analyzed the statistical importance. It's used Scikit-learn[1] Python implementation of Tree-based Classifiers to conduct this process. To determinate the K most relevant features, one must test multiple values of K, select the top features, and use them for training multiple models, and record their *F1-Scores*. Since the models need anomalous data to tune their hyperparameters and perform their final performance evaluation, samples of all the anomalous data sets were merged into a single dataset.

To test the efficiency of the chosen features, were tested for OC-SVM models with $K \in \{1, 5, 10, 15, 20, 25, 30, 35, 40, 45\}$ most relevant features, with 10 cross-validation splits. A OC-SVM is an unsupervised learning algorithm that's trained only on the "normal" data, in our case the negative examples. It learns the boundaries of those points and is, therefore, ready to classify any points that lie outside the boundary as, you guessed it, outliers. Training any quite unsupervised learning algorithm are often difficult and therefore the OC-SVM is not any exception. The nu parameter should be the proportion of outliers that we expect to watch (in our case we don't know, so we will test several magnitudes), the gamma parameter determines the smoothing of the contour lines. Therefore, the OC-SVM models used has the following combinations (table 4.1). All parameters not mentioned give worse results.

Combinations
gamma : 0.001, **kernel** : linear, **nu** : 0.001
gamma : 0.01, **kernel** : poly, **nu** : 0.01
degree : 1, **gamma** : auto, **kernel** : poly
gamma : 0.001, **nu** : 0.001
gamma : 0.001, **nu** : 0.01
gamma : 0.01, **kernel** : sigmoid, **nu** : 0.01
gamma : 0.01, **kernel** : sigmoid, **nu** : 0.001

Table 4.1: OC-SVM combinations for feature dimensionality.

[1]https://scikit-learn.org/stable/

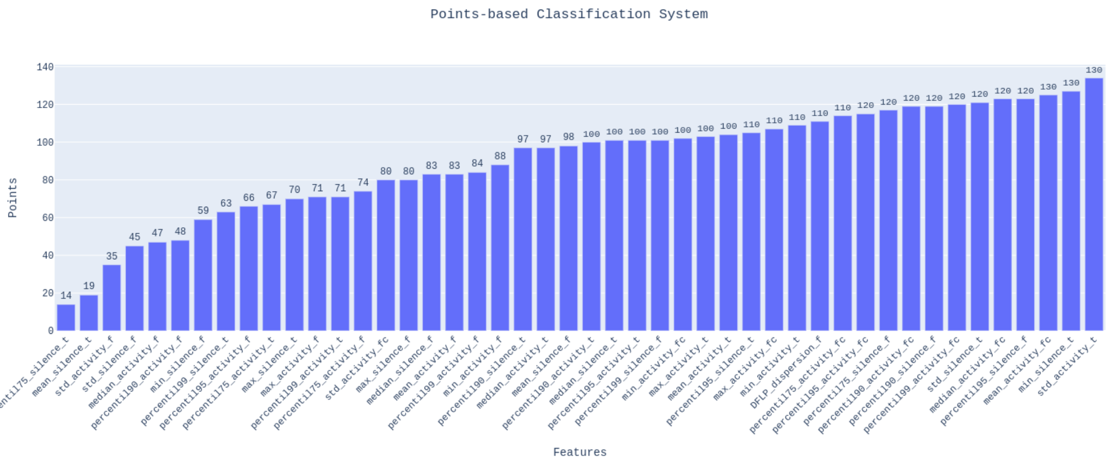
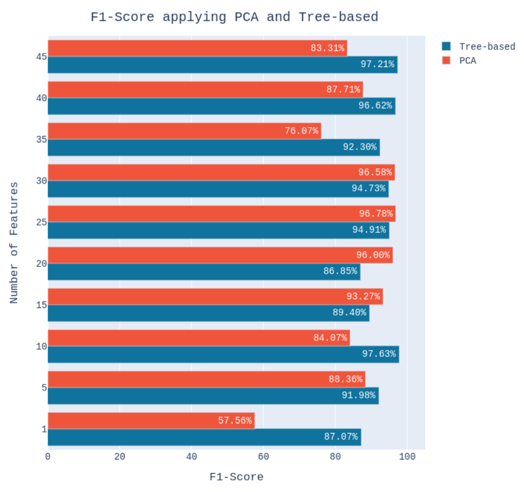


Figure 4.12: Relevance for each feature using Points-based Classification System

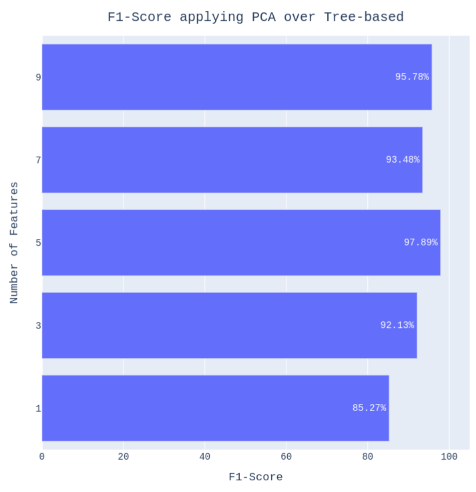
Given the number of features, we analyzed the importance of statistical data (figure 4.21). As said in section 3.6, features with few points are more relevant. The features are designated with a nomenclature (function)_(type of Feature)_(base information source). Each information source has the letters (f, fc and t), respectively for the active/inactive frequencies, consecutive active frequencies and activity/silence periods.

Taking this into account, it is possible to verify that the temporal features, more precisely the features based on silence are considered the most important, however the "podium" has a greater presence of features based on the number of active and inactive frequencies. On the contrary, frequencies based on consecutive active frequencies are not of great relevance for differentiation.

Now, it is possible to test the minimum number of features necessary to model the problem. It will be tested the influence of the features already ranked from the tree-based classifiers and the PCA on the model's result. To find a constant result and prove the robustness, the process was repeated 30 times for 10 cross-validation splits as was said earlier.



(a)



(b)

Figure 4.13: *F1-Score* for different number of features, using Tree-based Classifiers and PCA

There are 47 features in total. It would take time to study the performance for each total of features. Therefore, performance is studied for different numbers of features in order to understand how in which range of total features the best performance is found. Then the range of values before the maximum found in the previous study is studied in order to find the minimum total of features that contribute to performance.

Figure 4.13a show the minimum optimal number of characteristics is 10, using tree-based classifiers. The first 10 features are limited to features based on the number of active frequencies and features based on silence periods. Which goes according to the analysis made earlier. However, it was concluded that features based on the amount of active and inactive frequencies could lead to a large amount of false negatives, anomalous behavior to be considered normal. To guarantee this conclusion, different scenarios will be tested in the following chapter.

We also tested PCA for dimensionality reduction with a set {1, 5, 10, 15, 20, 25, 30, 35, 40, 45} components. It turns out that using PCA components as input features without using feature selection produces slightly worse results. The PCA needs more features to achieve a good result. If the number of features was not much higher, it was preferable to use this number of features since we would have more inputs for the classification. However, since the number of features used by the PCA is much higher, 10 features will be used.

Therefore, since the tree-based classifiers has better results than PCA, the PCA was applied to the result of the tree-based classifiers, seeking to obtain equal or better results using fewer features (figure 4.13b). In terms of processing and assuming that this methodology is applied on a large scale, the less features are used the better.

Defining the value of the tree-based classifiers and testing the use of PCA in the tree-based classifiers result features shows that there is a slight improvement in the result. The PCA allows a decrease in the number of features of 10 to 5 feature. It is important to note that we are dealing with an environment in which both anomalies and normal behavior were generated in a controlled manner, since it is a proof of concept as indicated at the beginning of this study. Therefore, the number of patterns considered by the selection of features will be reduced, which may explain the reduced number of features considered.

The features resulting from this scenario will be used in the next scenarios. Each scenario at the start has a different dataset and therefore, a feature selection must be applied to each dataset, however, due to the time that needs to be spent, it will not be done, using the features resulting from this scenario for the next scenarios.

4.1.3 Classification Results

The final step is to train each model with the selected features and to extract the results. The tested models were OC-SVM, IF, LOF, KDE, and GMM, using Scikit-learn [2] Python implementations. For each dataset and each model, the data was randomly split into 80% training and 20% testing five times ($\mu = 0.8, \tau = 5$) and split according K-Fold Cross-Validation method ten times ($\upsilon = 5$) for every tested set of hyperparameters. The batch size used for CV sets was equal to 30% of the training set ($\gamma = 0.3$), and so, 70% of the training set with "clean" data was used for training, while the remaining 30%, plus an equal percentage of outlier samples ($\rho = 0.5$), were used as CV data. Table 4.2 showcases which ranges of parameters were used for tuning the classical models. Values over or under the present ranges performed worse than the presented tested combinations.

Model	**Parameters ranges**
OC-SVM	`kernel`: poly, linear, sigmoid, `nu`: 0.01, 0.001, `gamma`: 0.01, 0.001, auto, `degree`: 1
IF	`max samples`: [1, 4999], `contamination`: [0.1,0.2]
LOF	`nn`: 5000, `contamination`: [0.1, 0.2]
KDE	`kernel`: `guassian`, `bandwidth`: [0.01, 0.1], `threshold`: 0.2
GMM	`n_components`: [2.0, 3.0], `covariance type`: full, `threshold`: 0.2

Table 4.2: Ranges of parameters used on classical OCC models tuning.

Since this scenario was considered a test run, we only considered classical models. Their performance does not scale well with higher-dimensional feature spaces. With this in mind, the input features vector corresponds to the K best features resulting from the feature selection, as exemplified in the previous section. Also, for the discussion of the results, when faced with two models of identical performance, the model with greater precision is predominating.

The 4.3 table shows that, OC-SVM was able to produce the best detection rates, with the cost of considerably short training times, averaging about 11 seconds to perform hyperparameter tuning using 10-fold cross-validation strategy. In addition, it also has short test times, averaging less than 1 second to classify about 5051 samples. IF were one of the fastest to converge, but the ones producing the worse classification results. The GMM and KDE seems to have excellent performance while maintaining reasonable training times. LOF has one of the least optimal detection rates and is about eight times slower to train than OC-SVM.

[2]https://scikit-learn.org/stable/

It was expected that there would be a greater number of false negatives, since the models are mono-class and they learn normal behavior. The identical data revealed in the data exploration section was expected to be considered false negatives. However, what happened was not exactly that. If we compare the precision and F1-Score values, the F1-Score values are not only affected by precision, but also by recall, so we are talking about the presence of both false positives and false negatives. In the case of OC-SVM in the study of "mixture of samples", the F1-Score result is completely affected by the number of false positives, since it is presented with a great precision. In all models, the confidence intervals were considerably high. What can justify this behavior is the difference between the scenarios tested. It is difficult to ensure the same types of behavior to be tested since the tests are done at home, with the use of the Internet being used frequently.

Type of anomaly		**OC-SVM**	**IF**	**LOF**	**KDE**	**GMM**
scheduled 120s	*Precision*	**100.0±3.74**	87.04±7.29	88.16±4.36	91.60±4.54	91.84±2.89
	F1-Score	**98.00±7.44**	93.07±6.69	93.04±8.90	88.92±5.00	89.20±0.94
scheduled 300s	*Precision*	**100.0±4.67**	73.88±5.73	73.79±6.35	91.72±0.98	91.35±1.04
	F1-Score	**100.0±4.34**	77.96±4.89	84.92±3.76	90.65±1.94	86.30±1.08
Mixture of samples	*Precision*	**100.0±7.36**	67.25±5.96	86.80±1.58	90.76±1.89	91.37±0.83
	F1-Score	**85.99±7.67**	80.42±6.77	88.13±1.99	84.54±1.06	89.98±1.17

Table 4.3: Home automation scenario classification results (in percentage, with a 95% confidence interval).

In sum, this was a simple scenario, since it only monitored three nodes maximum, one of them not licit. The usage of the spectrum may be one of the reasons for the results. The difficulty in maintaining a controlled environment may have influenced the results. Furthermore, the strategy chosen to face the problem was not the best. Even so, the models were able to maintain considerable performance in the dataset with normal background behavior.

Model	**Training time**	**Training set size**	**Testing time**	**Test set size**
OC-SVM	0min11s ± <1s	28128	**<1s**	5051.0±581
IF	0min3s ± <1s	28128	**<1s**	5051.0±581
LOF	1min27s ± 0min3s	28128	**<1s**	5051.0±581
KDE	0min4s ± 0min4s	28128	**<1s**	5051.0±581
GMM	0min3s ± <1s	28128	**<1s**	5051.0±581

Table 4.4: Average (with 95% confidence interval) training and testing durations in the home automation scenario.

4.2 IoT Home Scenario (with higher volume of data)

Contrasting with the scenario presented in section 4.1, which was initially thought as a bench test to validate the proposed methods in chapter 3. The following scenario showcases an attempt at detecting the same type of outlier behavior but in an environment with a higher number of nodes connected to the same AP and with Bluetooth devices, simulating a more realistic scenario. As the amount of non compromised connected devices is superior - in comparison to the maximum three in the previous case - the 2.4GHz spectrum will be far more often occupied, making it harder to detect the presence of periodic outlier behavior.

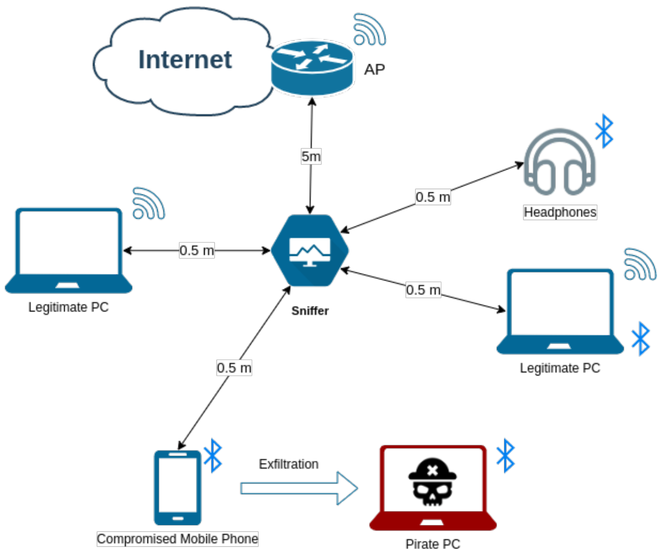


Figure 4.14: Home Scenario Network Diagram

The objective is also to identify a periodic outlier, within an environment containing interactions of two legitimate PCs with an AP, mostly YouTube videos, one of which is playing audio on Headphones. These iterations simulate normal traffic. Anomalous traffic is simulated in the same way as the previous scenario, with the exchange of files between the compromised mobile phone and the Pirate PC.

The data flow to simulate anomalous behavior maintains the same statistical distribution presented in the previous context, with a file transferred of 5 MB. The file transfer occurs every 120 and 300 seconds, referred to as *scheduled 120s* and *scheduled 300s*, respectively.

The HackRF LNA gain was set to 32dB, and the VGA gain to 16dB, just like in the previous case. It was placed about five meters away from the access point, and 0.5 meters from the adjacent devices, as described in the figure 4.14.

4.2.1 Data Exploration

This scenario will have about 1 hour of data per data set. Unlike the previous scenario, there were no restrictions on data capture. This scenario will have an identical environment to the previous one. The captures are made exclusively during the night to restrict the circulation of data on the network maintaining a controlled environment. As in the home automation scenario, all devices are connected to the same AP via channel 8, as in the previous scenario. All captured RSSI data were calculated at the frequencies of interest.

Figure 4.15 shows that there are some differences when compared with the same figure in the previous chapter (figure 4.3b). The activities seem to be less noticeable. It should be noted that the VGA value has a high value making the ground noise more noticeable in an environment with more devices, the latter being much higher due to the greater number of nodes and general background noise. The tests on the previous home automation scenario were carried out in a place where there was little interference in the 2.4 GHz band; thus, making it easier to discern the activity of soil noise. Despite having small activities all the time, only more significant data transfers are clear. In this scenario, there is the communication of the devices not only with AP, on the part of laptops, but via Bluetooth between laptop and headphones.

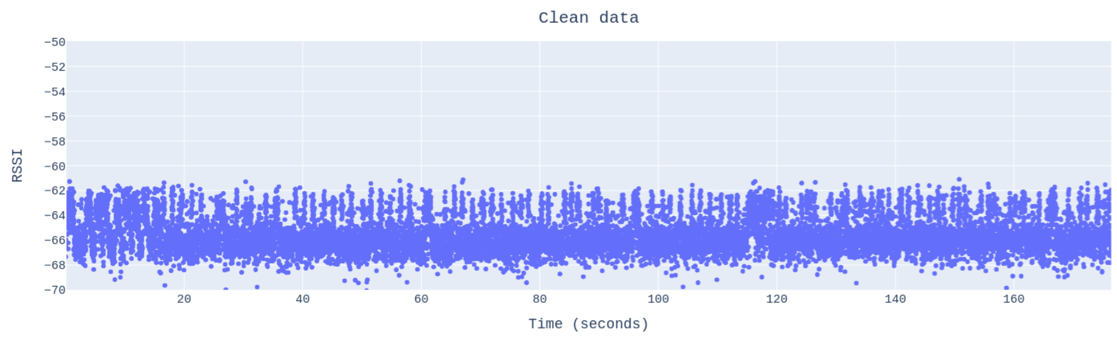


Figure 4.15: RSSI power values over time with several activities

Following the study, the threshold corresponding to the maximum of the values RSSI with no sign of relevant activity on the network was defined. Again, the data is divided into 10-minute sliding windows with a 5-second sliding distance, with a 1-second sampling window. In addition, activity/silence features, dispersion features and trading features are extracted from the time series data.

The exploration of the features is done by comparing a normal dataset with an anomalous dataset. The normal dataset contains uninterrupted data flow over Wi-Fi through the connection with the AP and Bluetooth, through the connection between Laptop and headphones and the anomalous dataset contains isolated anomalies but with normal behavior in the background. The intention is to verify what is highlighted with the addition of anomalies to the normal behavior of the network.

As for the first basic information source, which states that different communication technologies will reveal a total number of different active frequencies, it is no longer a valid premise for this scenario. In comparison with the previous scenario, a greater number of frequencies, not used by the previous scenario, is used as part of the normal behavior through the addition of Bluetooth devices (figure 4.16).

Regarding normal behavior, a more detailed analysis reveals that the number of inactive frequencies decreases, now maintaining a smaller range of the total active frequencies between 9 and 22. The most notable difference is in the average number of frequencies active by anomalous behavior. The increase average number of frequencies used complete overlap the average number of frequencies used by normal behavior. This reveals a concern already mentioned in the previous behavior. As in the previous scenario, this can lead to the possibility of false negatives. The use of the number of frequencies does not seem to be a good feature in differentiating behaviors.

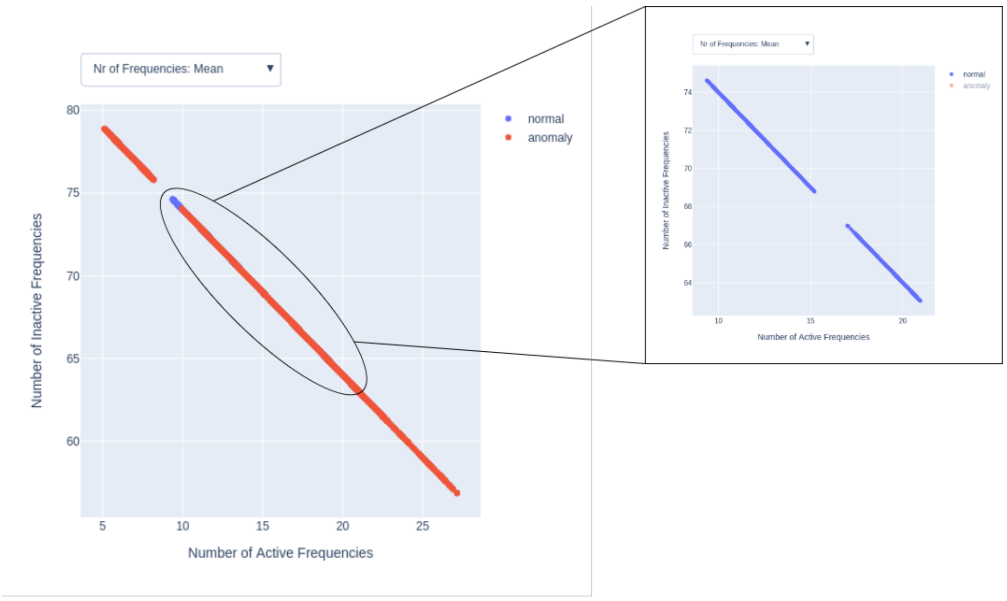


Figure 4.16: Comparison of the number of active and inactive frequencies in clean and outlier behaviors

Once again, looking at the 4.17 figure, you can see the similarity with the previous scenario. The maximum number of active frequencies remains very noticeable, despite the normal behavior of having Bluetooth connections. The 75th percentile shows that the number of active frequencies has remained within a small range, and the number of inactive frequencies decreases due to the presence of the Bluetooth connection, compared to the previous scenario. In addition, the anomalous behavior continues to present a set of data of similar characteristics, with the possibility of generating false negatives. The points when being classified as normal, the appearance of anomalies with identical statistics will lead to the wrong classification.

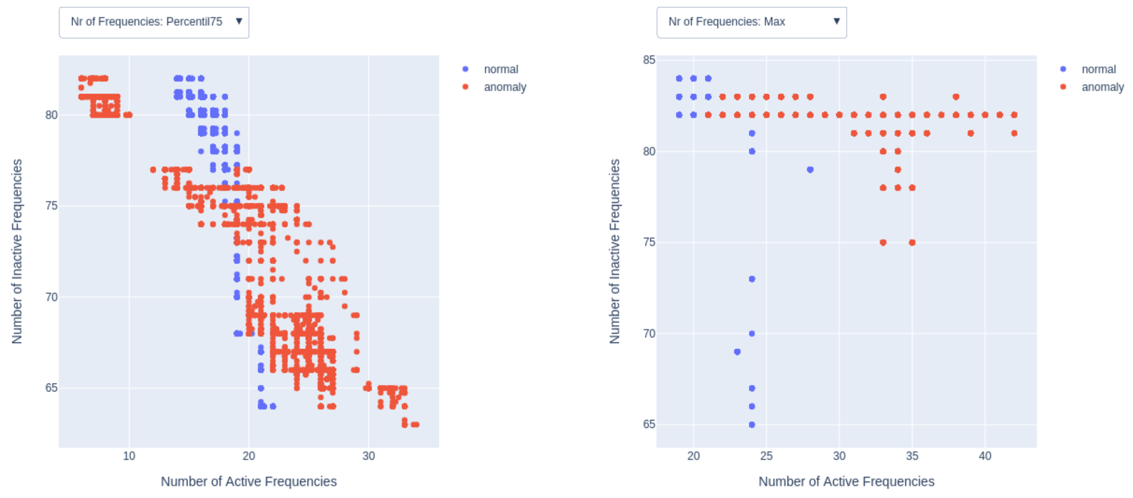


Figure 4.17: Percentile 75 and maximum number of frequencies in clean and outlier behaviors.

The DFLP indicator shows evidence of a behavior seen in the previous indicator. It was assumed that the addition of a Bluetooth connection to the normal behavior would increase the dispersion of active frequencies, however this is not true. The average number of active frequencies increased, but the dispersion of active frequencies did not have a major impact. On the other hand, the addition of another Bluetooth connection, which is the connection referring to the anomaly, placed this range with higher average values.

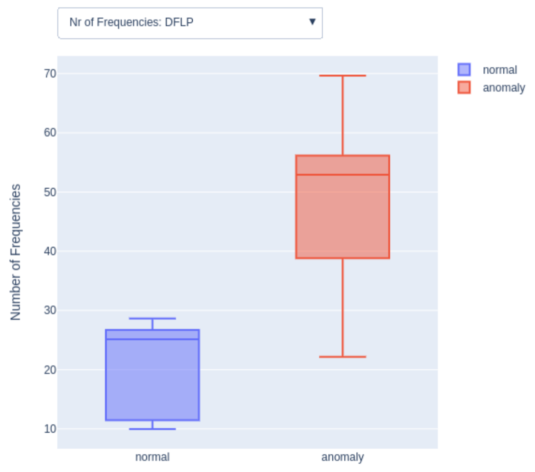
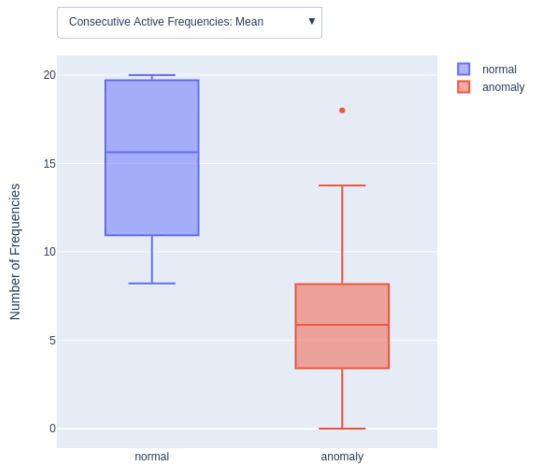


Figure 4.18: Mean difference between the first and the last active frequency in clean and outlier behaviors.

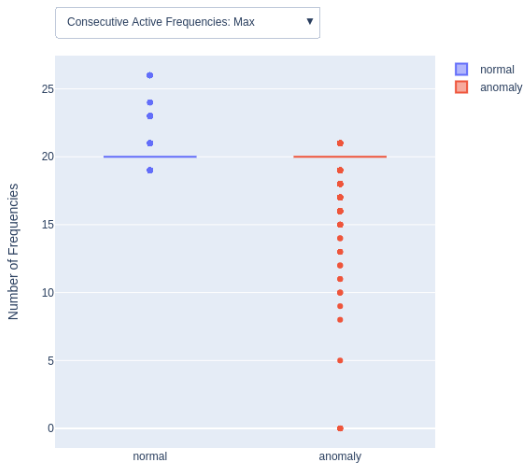
As mentioned in the previous scenario, the second assumption was based on the number of consecutive active frequencies. Since Wi-Fi uses fewer frequencies, then the number of consecutive frequencies will be greater than Bluetooth.

Unlike the previous scenario, the presence of more devices increases the number of consecutive frequencies (figure 4.19a). A behavior quite different from the previous one where it presented similar values between normal and anomalous behavior. An intriguing observation is the fact that with the presence of anomalies the number of consecutive frequencies decreases. One possible explanation is the number of devices using the spectrum. Eventually, increasing the number of devices, the use of the spectrum will increase, consequently, the number of active frequencies and the number of consecutive active frequencies. However, there are few devices on the network, and therefore the addition of a new Bluetooth communication does not increase the consecutive active frequencies, but rather increases the dispersion that was initially generated by the presence of a single Bluetooth connection.

In the previous scenario, the maximum values supported the assumption, in which the difference in maximum values for the number of consecutive frequencies was notorious, but in this case the maximum values of the behaviors remained identical, passing from a normal distribution of median 20 for normal behavior, to a constant value 20.



(a) Average number of consecutive frequencies.



(b) Maximum number of consecutive frequencies.

Figure 4.19

The average time proved to be a promising factor for the differentiation of behaviors, prevailing even with the addition of a Bluetooth connection. Analyzing this case, long periods of activity prevail in the normal scenario, but mainly periods of silence continue to be the determining factor in distinguishing behaviors (figure 4.20).

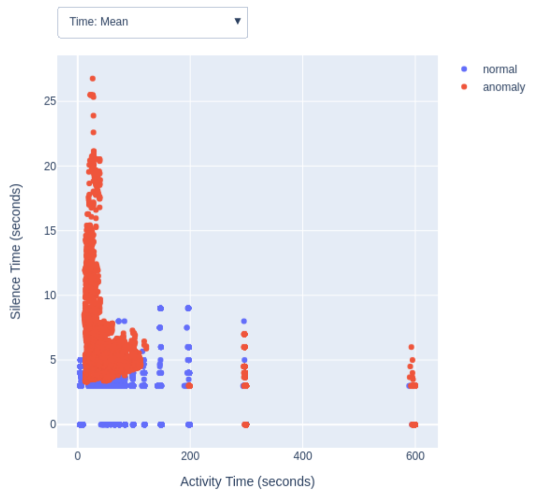


Figure 4.20: Comparison of the mean silence and activity periods on clean and outlier behaviors

In sum, eventually with the increase in devices, regardless of the communication technology they use, the spectrum will be so full that the indicators based on the number of frequencies will be irrelevant. However in a controlled environment, for example, an IoT environment, in which the devices have a static position and periodic activities, indicators based on the number of frequencies may still be useful to some extent. The usefulness of indicators based on active frequencies depends on the context of the use of the sniffer, which in terms of production is not feasible, as the anomaly detection method has to encompass the largest number of scenarios.

4.2.2 Feature Dimensionality

The next step was to perform an analysis of the relevance of input features. Besides using the feature selection mechanisms proposed in section 3.5, we also test feature dimensionality reduction techniques such as PCA. Similarly to the previous example, we used Scikit-learn[3] Python implementation of Tree-based Classifiers and PCA to conduct this process. For any subset of features, the number of cross-validation splits was 10, and for both the number of relevant features and PCA components, $K \in \{1, 5, 10, 15, 20, 25, 30, 35, 40, 45\}$ set.

The increase in the number of connections foresaw an increase in the dependence on temporal resources, however this did not happen. On the contrary, it showed that the podium is shared exclusively by features based on the number of active frequencies. A possible explanation, given previously, is the not strong use of the spectrum. Despite the increase in the number of connections, it is not enough to disable features based on active and inactive frequencies.

[3]https://scikit-learn.org/stable/

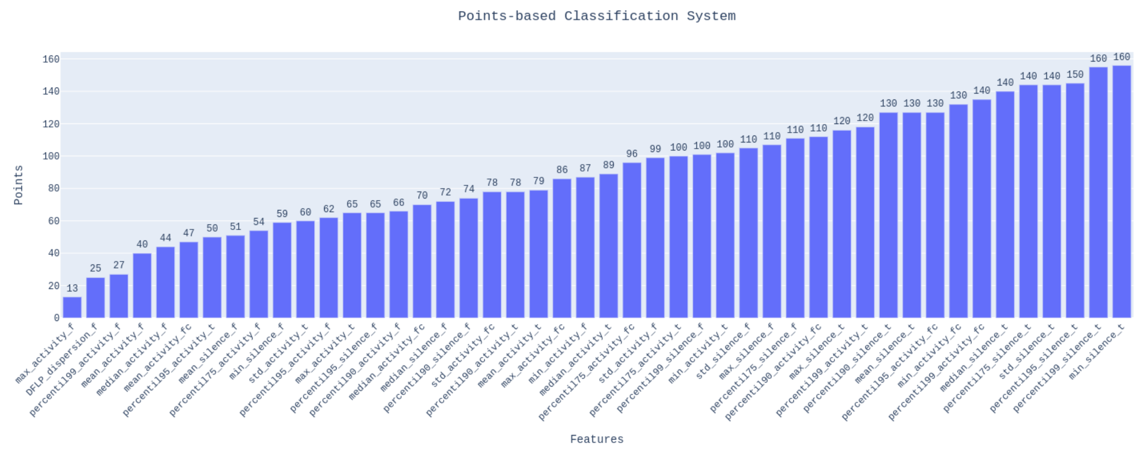
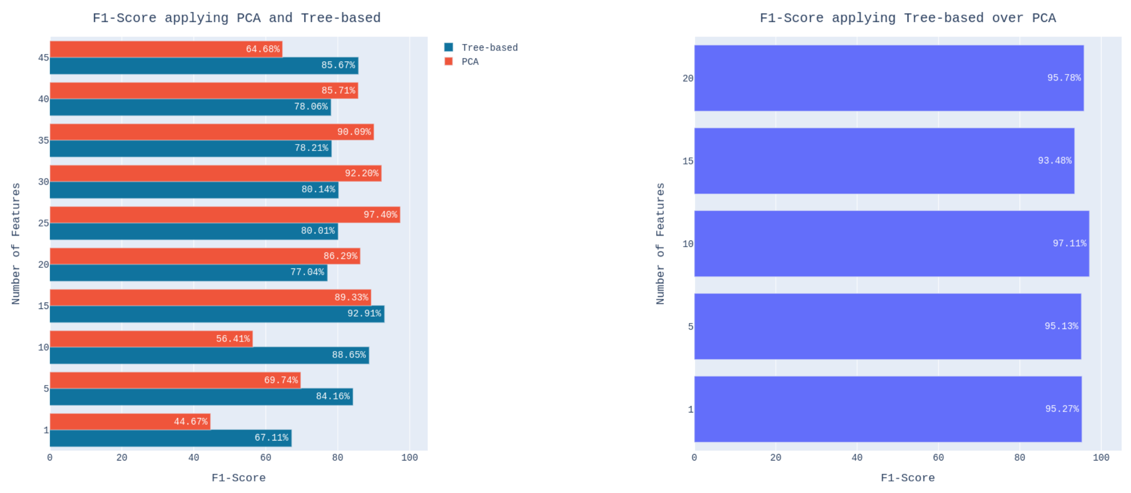


Figure 4.21: Relevance for each feature using Points-based Classification System

Figure 4.22a shows that, compared to resource selection, the PCA performs better. On the other hand, in relation to resource selection methods, the use of 10 or 15 resources in data sets produces excellent results but below the optimum score produced by the PCA. The application of tree-based classifiers on the PCA results did not bring any added value. The results showed slightly worse or almost equal. Although it can increase training time, we chose to use 25 features, as the extra information can be useful for other models not tested so far, such as KDE or GMM.



(a) (b)

Figure 4.22: *F1-Score* for different number of features, using Tree-based Classifiers and PCA

4.2.3 Classification Results

The data were randomly split into 80% training, and 20% testing for each of five conducted training procedures on every model. For classical models, it was then split according to the K-fold cross-validation method ten times for every set of hyperparameters, with a batch size equal to 30% of the training set, and a 50% rate of outlier samples in the CV data. Identical to the previous scenario, we resorted to Scikit-learn implementations. The parameters used to tune the models were the same as in the previous scenarios (table 4.2).

As shown in table 4.5, and identical to the previous scenario, OC-SVM outperforms all other models, obtaining the highest detection score. Contrary to the results obtained in the previous scenario, the LOF had a better performance, almost as good as the OC-SVM, however with a longer training time. OC-SVM has both a high precision rate and a large F1-Score, showing excellent behavior in identifying normal and abnormal behaviors. The IF model has lower F1-Score scores that come from lower precision rate, which means that they sometimes produce false positives, although they are able to identify almost all anomalies (high recall). The rest of the models have high F1-Score rates accompanied by equivalent precision, which means that they produce false negatives. In this scenario, the range of values remains high and as such the possible justification remains the same as in the previous scenario. What can justify this behavior is the difference between the scenarios tested. It is difficult to ensure the same types of behavior to be tested since the tests are done at home, with the use of the Internet being used frequently.

Type of anomaly		**OC-SVM**	**IF**	**LOF**	**KDE**	**GMM**
scheduled 120s	*Precision*	**99.42±1.47**	88.74±6.22	98.67±9.38	95.13±4.10	91.11±0.81
	F1-Score	**99.45±5.41**	94.03±10.02	99.33±7.51	93.25±2.11	85.03±2.07
scheduled 300s	*Precision*	**100.0±5.16**	75.71±7.50	95.64±4.62	89.52±2.79	85.66±2.74
	F1-Score	**93.00±8.33**	86.18±5.87	97.36±9.79	88.06±2.67	84.30±2.07

Table 4.5: Home automation scenario classification results (in percentage, with a 95% confidence interval).

In sum, classical models proved that, even in scenarios with a higher number of devices and increased background noise, they are still capable of discerning outlier behavior, since they maintained results identical to the tests carried out between normal and exclusively abnormal behavior.

Model	**Training time**	**Training set size**	**Testing time**	**Test set size**
OC-SVM	0min3s ± <1s	12237	**<1s**	2578.0±405
IF	0min3s ± <1s	12237	**<1s**	2578.0±405
LOF	0min22s ± 0min1s	12237	**<1s**	2578.0±405
KDE	0min1s ± <1s	12237	**<1s**	2578.0±405
GMM	0min1s ± <1s	12237	**<1s**	2578.0±405

Table 4.6: Average (with 95% confidence interval) training and testing durations in the home automation scenario.

4.3 Conclusion

Realistically, it's not possible to check every type of existing anomalies. This work covered only a few variations, more specifically, small volume periodic behavior, simulating an attacker exfiltrating information on a regular basis. Anomalies with a greater volume of knowledge exchange would be more easily detected because the respective periods of activity would be relatively long and the average number of active frequencies will be higher, however it is not possible to make the same conclusion for corporate networks due to the large flow of information caused by other devices, but it is possible to assume that for IoT networks this is possible due to the low volume of information exchanged between IoT devices. Obviously, if an attacker is aware of the periodic behavior of the network, it would be easier to masquerade the outlier behavior; yet, we are assuming that an external entity does not have that information. Additionally, the attacker may disguise the anomaly by mimicking the typical operation of a specific device, if he was aware that such a node existed in the network. Even if he managed to do it, he would only mimic the device bot-like behavior, not the human interactions that also describe its usual operation.

Moreover, these tests were conducted in a controlled environments. The home scenario occurred in a location with reduced number of devices connected to the network and little background noise or interference, since it is difficult to carry out the experiments when people at home are using the same network. The work environment tests were always conducted at the same period of the day - usually between 0 and 6 a.m. - and the agents connected to the network were always the same, despite slight variations on their behavior. Besides, the amount of data available was relatively low. In order to create a more reliable model, we would have to record a lot more training data, as well as more diverse.

CHAPTER 5

Conclusions and Future Work

Although this dissertation covered the base approach to allow the detection of anomalies at the physical layer, there is still a few points where it can be improved.

5.1 Conclusions

The study in Chapter 2 shows that IoT environments are susceptible to a spread of attacks, with the IoT environment increasing the attack surface for remote and physical attacks. In many cases, like IoT devices, they're shipped from the factory in a vulnerable state and have never been fixed or have implemented trivial security mechanisms. Sometimes, it's physically impossible to repair these devices.

To overcome these issues, administrators were forced to use monitoring strategies to keep networks secure, although at the same time they're disrespecting users' privacy as they're ready to correlate network patterns to an individual node. On the opposite hand, the proposed anomaly detection mechanisms specialize in physical layer data, where such a task becomes more complex because the signal mixes multiple devices data exchanges.

The mentioned mechanisms rely on analyzing and extracting statistical and time-frequency features that characterize radio silence/activity periods and active frequencies. Moreover, the goal is to only model the network when it's freed from anomalies, in order that the trained classifier isn't bound to any specific form of outlier behavior. Another issue of current anomaly detection systems in IoT networks is that the majority rely on supervised learning techniques, thus training a classifier to specific forms of attack vectors, i.e. signature-based. With this in mind, we proposed a series of feature engineering procedures to optimize the classification performance of OCC models. Our choice for one-class classifiers is justified by only having to train them with "clean" data, therefore, avoiding biasing them towards some determined anomalous behavior. This also made it possible to make a more robust model against unseen phenomena.

Furthermore, as we suggested the utilization of SDRs because the primary tool to record IQ data, these mechanisms aren't bound to any specific radio technology, and that they are often used with Wi-Fi, Bluetooth, or any other that works on the frequency range supported by the provided sniffer.

To validate those same mechanisms to outlier detection at the physical layer, we designed a home network scenario, where the goal was to detect periodic outlier behavior in a Bluetooth channel working as a proof-of-concept scene.

The OCC classical models were able to detect the anomalous behavior with some difficulties. Some models were able to obtain between 75% e 99% detection rate. One positive note about the developed models is that the majority also presented satisfactory precision rates.

These tests showed the validity of the mechanisms proposed in chapter 3, although it's realistically impossible to check all kinds of existing anomalies. We only tested low-volume and considerably high-frequency behavior, because it seemed harder to detect smaller data transfers because the remaining traffic masked them. The tests were always conducted with the same devices, with similar behaviors whenever, at an equivalent periods of the day. This makes it harder for the model to generalize to other periods of the day where there would be more connected devices or maybe differences in behavior between weekdays and weekends.

5.2 Future Work

Despite what this Dissertation addressed, there are several points that can be improved and others that can be added:

- In our test scenarios, only short-length and high-frequency anomalies were considered. It is impossible to test all imaginable variations of anomalous behavior, especially since it depends on the regular operation of the network. Nevertheless, having a longer list of anomalies, particularly long-duration and low-frequency ones (e.g., uploading a 1GB video every ten minutes) increases the chances of creating a model that is properly tuned.
- In our scenarios, periodic tests were considered, however IoT networks, in addition to periodic and automated behaviors, can have human or asynchronous interactions, that is, non-periodic. Making tests considering these iterations are important, mainly imitating legal behaviors.
- Our tests were conducted with a few IoT devices. It would be interesting to study the network behavior with more devices and how they work, and see there impact on the spectrum usage. It will be more challenging understand and model such behaviors correctly.
- The pre-processing methodology presented in chapter 4, a threshold was used to isolate the relevant data, but it was interesting, a priori, to use a **EMD**. It aims to decompose a signal into elemental ones, called Intrinsic Mode Functions (IMFs). The first Intrinsic Mode Function (IMF) usually carries the most oscillating (high-frequency) components, it can be rejected to remove high-frequency components (e.g., random noise) [127].

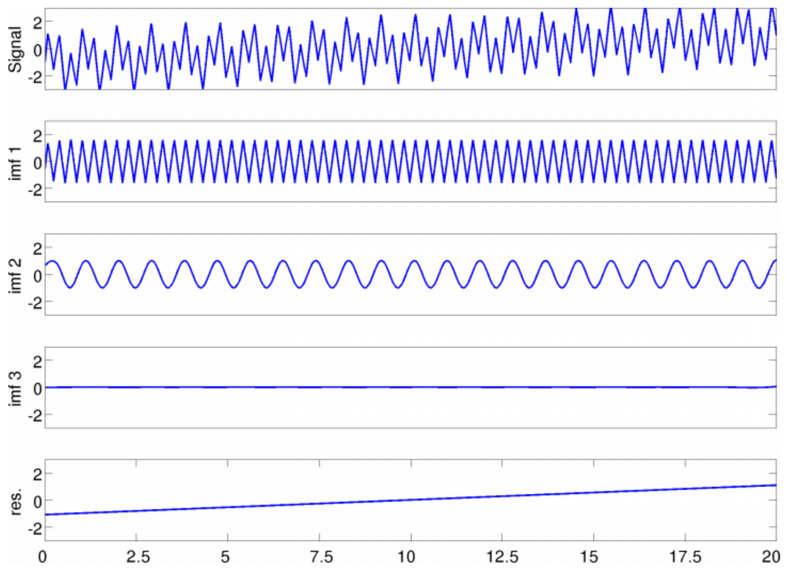


Figure 5.1: Illustration of an EMD decomposition of a toy signal composed of a sawtooth, a sinusoid and a linear trend (retrieved from [127])

The intention of this pre-processing would be to try to isolate the relevant waves, instead of relying on a threshold to isolate the network activity. It should be remembered that the results detected activity at unused frequencies, pointing out that just applying a threshold will not be enough.

- One of the problems present at the end of the tests was the presence of false positives. A possible solution to be tested for the reduction of false negatives is the implementation of a second level of classification, which would involve decision making based on more than one window. When considering N windows, if one of them was anomalous, all the others would be considered anomalous. This solution would not have a great impact on the methodology followed, since the sliding of the observation windows is only 5 seconds. So instead of making the decision after 10 minutes (time from an observation window), the decision was made after 10 minutes * N * 5 seconds.
- The business scenario addressed initially mentions a system for sending data collected by sniffers, allocated at strategic points, to a central entity. In this scenario, the sniffer is no longer a passive element, but an active element as it introduces data into the network. It would be interesting to study the network with the introduction of this data, in order to draw the conclusion whether or not it is possible to use the existing network to send the data or it will be necessary to have an exclusive network parallel to the existing network for sending data.

References

[1] ESET, “Threat report q2 2020”, WeLiveSecurity, Tech. Rep., 2020, p. 29. [Online]. Available: https://www.eset.com/fileadmin/ESET/CZ/Threat-report/Q2-2020_Threat_Report.pdf (visited on 01/08/2020).

[2] X. Liu, Y. Liu, A. Liu, and L. T. Yang, “Defending on–off attacks using light probing messages in smart sensors for industrial communication systems”, *IEEE Transactions on Industrial Informatics*, vol. 14, no. 9, pp. 3801–3811, 2018.

[3] M. Hasan, M. M. Islam, M. I. I. Zarif, and M. Hashem, “Attack and anomaly detection in iot sensors in iot sites using machine learning approaches”, *Internet of Things*, vol. 7, p. 100 059, 2019. DOI: 10.1016/j.iot.2019.100059.

[4] “How much would a data breach cost your business?”, IBM, Tech. Rep., 2020. [Online]. Available: https://www.ibm.com/security/digital-assets/cost-data-breach-report/#/pt (visited on 01/10/2020).

[5] J. Desjardins, “The 15 biggest data breaches in the last 15 years”, *Visual Capitalist*, Jun. 2019. [Online]. Available: https://www.visualcapitalist.com/the-15-biggest-data-breaches-in-the-last-15-years/ (visited on 01/10/2020).

[6] F. Ullah, M. Edwards, R. Ramdhany, R. Chitchyan, M. A. Babar, and A. Rashid, “Data exfiltration: A review of external attack vectors and countermeasures”, *Journal of Network and Computer Applications*, vol. 101, pp. 18–54, 2018.

[7] J. Hernantes, G. Gallardo, and N. Serrano, “It infrastructure-monitoring tools”, *IEEE Software*, vol. 32, no. 4, pp. 88–93, 2015. DOI: 10.1109/MS.2015.96.

[8] I. B. Brasil, *Entendendo os conceitos entre os modelos tcp/ip e osi*, Oct. 2019. [Online]. Available: https://www.iperiusbackup.net/pt-br/entendendo-os-conceitos-entre-os-modelos-tcpip-e-osi/ (visited on 01/18/2020).

[9] R. Sommer and V. Paxson, “Outside the closed world: On using machine learning for network intrusion detection”, *2010 IEEE Symposium on Security and Privacy*, 2010. DOI: 10.1109/sp.2010.25.

[10] D. E. Denning, “An intrusion-detection model”, *IEEE Transactions on software engineering*, no. 2, pp. 222–232, 1987.

[11] H. S. Javitz, A. Valdes, and C. NRaD, “The nides statistical component: Description and justification”, *Contract*, vol. 39, no. 92-C, p. 0015, 1993.

[12] W. Lee and D. Xiang, “Information-theoretic measures for anomaly detection”, in *Proceedings 2001 IEEE Symposium on Security and Privacy. S&P 2001*, IEEE, 2000, pp. 130–143.

[13] Z. Zhang, J. Li, C. Manikopoulos, J. Jorgenson, and J. Ucles, “Hide: A hierarchical network intrusion detection system using statistical preprocessing and neural network classification”, in *Proc. IEEE Workshop on Information Assurance and Security*, 2001, pp. 85–90.

[14] W. Hu, Y. Liao, and V. R. Vemuri, “Robust anomaly detection using support vector machines”, in *Proceedings of the international conference on machine learning*, Citeseer, 2003, pp. 282–289.

[15] C. Sinclair, L. Pierce, and S. Matzner, “An application of machine learning to network intrusion detection”, in *Proceedings 15th Annual Computer Security Applications Conference (ACSAC’99)*, IEEE, 1999, pp. 371–377.

[16] K. Rose, S. Eldridge, and L. Chapin, "The internet of things: An overview", *The internet society (ISOC)*, vol. 80, pp. 1–50, Oct. 2015. [Online]. Available: https://www.internetsociety.org/resources/doc/2015/iot-overview.

[17] *A internet das coisas (iot – internet of things)*. [Online]. Available: https://www.cncs.gov.pt/a-internet-das-coisas-iot-internet-of-things/ (visited on 03/03/2020).

[18] T. Huelsman, E. Powers, S. Peasley, and R. Robinson, "Cyber risk in advanced manufacturing", 2016. [Online]. Available: https://www2.deloitte.com/content/dam/Deloitte/us/Documents/manufacturing/us-manufacturing-cyber-risk-in-advanced-manufacturing-executive-summary.pdf.

[19] D. Takahashi, K. Wiggers, P. Sawers, and N. Davies, "Softbank believes 1 trillion connected devices will create $11 trillion in value by 2025", *VentureBeat*, 2020. [Online]. Available: https://venturebeat.com/2018/10/16/softbank-believes-1-trillion-connected-devices-will-create-11-trillion-in-value-by-2025/.

[20] K. Pequenino, *Pandemia esta a originar o maior volume deciberataques que ja vimos*, Mar. 2020. [Online]. Available: https://www.publico.pt/2020/03/31/tecnologia/noticia/pandemia-originar-maior-volume-ciberataques-ja-vimos-1910028 (visited on 03/17/2020).

[21] A. Ng, "Iot devices need built-in security standards, ul says", *CNET*, Dec. 2019. [Online]. Available: https://www.cnet.com/news/iot-devices-need-security-standards-built-in-ul-says/ (visited on 03/21/2020).

[22] R. Ashley, "Statement for the record: Worldwide threat assessment", *Defense Intelligence Agency*, Mar. 2018. [Online]. Available: https://www.dia.mil/News/Speeches-and-Testimonies/Article-View/Article/1457815/statement-for-the-record-worldwide-threat-assessment/.

[23] O. Security Team, *Owasp internet of things top 10 2018*, 2018. [Online]. Available: https://owasp.org/www-pdf-archive/OWASP-IoT-Top-10-2018-final.pdf (visited on 03/21/2020).

[24] "The case for automotive gnss cyber security measures", *Regulus.com*, Feb. 2020. [Online]. Available: https://www.regulus.com/blog/the-case-for-automotive-gnss-cyber-security-measures (visited on 03/22/2020).

[25] A. Nogueira, "Falha em segurança em brinquedos eletrónicos expõe milhares de crianças", *Security Information News*, 2015. [Online]. Available: https://securityinformationnews.com/2015/12/01/falha-em-seguranca-em-brinquedos-eletronicos-expoe-milhares-de-criancas/.

[26] M. Kuzlu, M. Pipattanasomporn, and S. Rahman, "Review of communication technologies for smart homes/building applications", *2015 IEEE Innovative Smart Grid Technologies - Asia (ISGT ASIA)*, 2015. DOI: 10.1109/isgt-asia.2015.7437036.

[27] Maravedis, "The role of wifi in iot", *IoT For All*, Mar. 2020. [Online]. Available: https://www.iotforall.com/wifi-role-iot (visited on 04/03/2020).

[28] R. Triggs, "A quick history of bluetooth", *Android Authority*, Mar. 2018. [Online]. Available: https://www.androidauthority.com/history-bluetooth-explained-846345/ (visited on 04/09/2020).

[29] C. Mathias, "Bluetooth is dead", *EETimes*, Oct. 2003. [Online]. Available: https://www.eetimes.com/bluetooth-is-dead/ (visited on 04/09/2020).

[30] D. Hollander, "The state of bluetooth in 2018 and beyond", *Bluetooth Technology Website*, Apr. 2018. [Online]. Available: https://www.bluetooth.com/blog/the-state-of-bluetooth-in-2018-and-beyond/ (visited on 05/13/2020).

[31] "Understanding bluetooth range", *Bluetooth Technology Website*, [Online]. Available: https://www.bluetooth.com/learn-about-bluetooth/bluetooth-technology/range/ (visited on 05/20/2020).

[32] J. Arellano, "Bluetooth vs. wi-fi for iot: Which is better?", *verypossible.com*, Jul. 2019. [Online]. Available: https://www.verypossible.com/insights/bluetooth-vs.-wi-fi-for-iot-which-is-better (visited on 05/13/2020).

[33] I. Chatzigiannakis, "19 - apps for smart buildings: A case study on building security", in *Start-Up Creation*, F. Pacheco-Torgal, E. Rasmussen, C.-G. Granqvist, V. Ivanov, A. Kaklauskas, and

S. Makonin, Eds., Woodhead Publishing, 2016, pp. 465–479, ISBN: 978-0-08-100546-0. DOI: https://doi.org/10.1016/B978-0-08-100546-0.00019-4. [Online]. Available: http://www.sciencedirect.com/science/article/pii/B9780081005460000194.

[34] K.-H. Chang, "Bluetooth: A viable solution for iot? [industry perspectives]", *IEEE Wireless Communications*, vol. 21, no. 6, pp. 6–7, 2014. DOI: 10.1109/mwc.2014.7000963.

[35] A. K. Sood and R. Enbody, "Chapter 1 - introduction", in *Targeted Cyber Attacks*, A. K. Sood and R. Enbody, Eds., Boston: Syngress, 2014, pp. 1–9, ISBN: 978-0-12-800604-7. DOI: https://doi.org/10.1016/B978-0-12-800604-7.00001-2. [Online]. Available: http://www.sciencedirect.com/science/article/pii/B9780128006047000012.

[36] M. Smith, "U.s. government is 'biggest buyer' of zero-day vulnerabilities, report claims", *CSO Online*, May 2013. [Online]. Available: https://www.csoonline.com/article/2224620/u-s--government-is--biggest-buyer--of-zero-day-vulnerabilities--report-claims.html (visited on 05/15/2020).

[37] J. Porup, "What is a zero day? a powerful but fragile weapon", *CSO Online*, 2019. [Online]. Available: https://www.csoonline.com/article/3284084/what-is-a-zero-day-a-powerful-but-fragile-weapon.html.

[38] T. Yadav and A. M. Rao, "Technical aspects of cyber kill chain", *Communications in Computer and Information Science*, pp. 438–452, 2015. DOI: 10.1007/978-3-319-22915-7_40.

[39] A. K. Sood and R. J. Enbody, "Targeted cyberattacks: A superset of advanced persistent threats", *IEEE Security Privacy*, vol. 11, no. 1, pp. 54–61, 2013. DOI: 10.1109/MSP.2012.90.

[40] P. Engebretson, "Chapter 2 - reconnaissance", in *The Basics of Hacking and Penetration Testing*, P. Engebretson, Ed., Boston: Syngress, 2011, pp. 15–41, ISBN: 978-1-59749-655-1. DOI: https://doi.org/10.1016/B978-1-59749-655-1.00002-7. [Online]. Available: http://www.sciencedirect.com/science/article/pii/B9781597496551000027.

[41] L. D. B. Macias Mortier, "Steps in the cyber kill chain", *CybelAngel*, [Online]. Available: https://cybelangel.com/blog/steps-in-the-cyber-kill-chain/ (visited on 05/20/2020).

[42] "Verizon: Data breach investigations report 2020", *Computer Fraud Security*, vol. 2020, no. 6, p. 4, 2020, ISSN: 1361-3723. DOI: https://doi.org/10.1016/S1361-3723(20)30059-2. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1361372320300592.

[43] "Internet security threat report", Symantec, 350 Ellis Street, Mountain View, CA 94043, United Stated of America, Tech. Rep., Feb. 2019. [Online]. Available: https://docs.broadcom.com/doc/istr-24-2019-en.

[44] Y. Said, "Steps of the cyber kill chain - linux hint", *linuxhint.com*, 2020. [Online]. Available: https://linuxhint.com/cyber_kill_chain_steps/ (visited on 05/22/2020).

[45] N. Villeneuve, T. Haq, and N. Moran, *Evasive tactics: Taidoor*, 2013. [Online]. Available: https://www.fireeye.com/blog/threat-research/2013/09/evasive-tactics-taidoor-3.html.

[46] M. A. F. Company, *M-Trends 2015 : A view from the front lines*. 2015. [Online]. Available: http://www2.fireeye.com/rs/fireye/images/rpt-m-trends-2015.pdf.

[47] "What is a data exfiltration?", *Infoblox*, [Online]. Available: https://www.infoblox.com/glossary/data-exfiltration/ (visited on 05/25/2020).

[48] S. Ikeda, *Ransomware attack on portuguese energy company edp shows increasing trend toward public leaking of sensitive information*, Apr. 2020. [Online]. Available: https://www.cpomagazine.com/cyber-security/ransomware-attack-on-portuguese-energy-company-edp-shows-increasing-trend-toward-public-leaking-of-sensitive-information/ (visited on 05/25/2020).

[49] C. J. D'Orazio, K.-K. R. Choo, and L. T. Yang, "Data exfiltration from internet of things devices: Ios devices as case studies", *IEEE Internet of Things Journal*, vol. 4, no. 2, pp. 524–535, 2017. DOI: 10.1109/jiot.2016.2569094.

[50] A. Giani, V. H. Berk, and G. V. Cybenko, "Data exfiltration and covert channels", in *Sensors, and Command, Control, Communications, and Intelligence (C3I) Technologies for Homeland Security and Homeland Defense V*, International Society for Optics and Photonics, vol. 6201, 2006, p. 620 103.

[51] "10 steps to cyber security - monitoring", *ncsc.gov.uk*, Jan. 2019. [Online]. Available: `https://www.ncsc.gov.uk/collection/10-steps-to-cyber-security/the-10-steps/monitoring` (visited on 05/29/2020).

[52] "Report itu-r sm.2355-1 - spectrum monitoring evolution", International Telecommunication Union, Tech. Rep., Jun. 2019. [Online]. Available: `https://www.itu.int/dms_pub/itu-r/opb/rep/R-REP-SM.2355-1-2019-PDF-E.pdf`.

[53] K. Gold, "Active vs passive network monitoring", *EXFO*, Feb. 2019. [Online]. Available: `https://www.exfo.com/en/resources/blog/active-passive-network-monitoring/` (visited on 06/03/2020).

[54] S. Ligus, "Chapter 2 - monitoring", in *Effective Monitoring and Alerting*, O'Reilly, 2012, pp. 15–47, ISBN: 9781449333522. [Online]. Available: `https://books.google.com.br/books?id=KirJSqFcWIEC`.

[55] M. Ã. Sastoque-Caro, G. A. Puerto-LeguizamÃ, and C. A. SuÃ¡rez-Fajardo, "Opportunities to implement Software Defined Radio in network sensors", en, *Revista Facultad de IngenierÃa*, vol. 26, pp. 137–148, Aug. 2017, ISSN: 0121-1129. [Online]. Available: `http://www.scielo.org.co/scielo.php?script=sci_arttext&pid=S0121-11292017000200137&nrm=iso`.

[56] A. Gupta, "Software defined radio", in *The IoT Hacker's Handbook: A Practical Guide to Hacking the Internet of Things*. Berkeley, CA: Apress, 2019, pp. 223–263, ISBN: 978-1-4842-4300-8. DOI: `10.1007/978-1-4842-4300-8_9`. [Online]. Available: `https://doi.org/10.1007/978-1-4842-4300-8_9`.

[57] V. Bhuse and A. Gupta, "Anomaly intrusion detection in wireless sensor networks", *Journal of High Speed Networks*, vol. 15, no. 1, pp. 33–51, 2006. [Online]. Available: `https://www.researchgate.net/profile/Haiguang_Chen/publication/221451424_Lightweight_Anomaly_Intrusion_Detection_in_Wireless_Sensor_Networks/links/567f4ecf08ae19758389a897/Lightweight-Anomaly-Intrusion-Detection-in-Wireless-Sensor-Networks.pdf`.

[58] M. Zhang, A. Raghunathan, and N. K. Jha, "Medmon: Securing medical devices through wireless monitoring and anomaly detection", *IEEE Transactions on Biomedical Circuits and Systems*, vol. 7, no. 6, pp. 871–881, 2013. DOI: `10.1109/TBCAS.2013.2245664`.

[59] A. Rashid, H. Chivers, G. Danezis, E. Lupu, A. Martin, Y. Rigby, and J. Hallett, *CyBOK*, 1st ed. 2019, pp. 572–578.

[60] "The pros and cons of intrusion detection systems", *Rapid7 Blog*, Jan. 2017. [Online]. Available: `https://blog.rapid7.com/2017/01/11/the-pros-cons-of-intrusion-detection-systems/` (visited on 06/20/2020).

[61] T. T. Nguyen and G. Armitage, "A survey of techniques for internet traffic classification using machine learning", *IEEE Communications Surveys Tutorials*, vol. 10, no. 4, pp. 56–76, 2008. DOI: `10.1109/surv.2008.080406`.

[62] F. Sabahi and A. Movaghar, "Intrusion detection: A survey", *Systems and Networks Communication, International Conference on*, vol. 0, pp. 23–26, Jan. 2008. DOI: `10.1109/ICSNC.2008.44`.

[63] N. Chaabouni, M. Mosbah, A. Zemmari, C. Sauvignac, and P. Faruki, "Network intrusion detection for iot security based on learning techniques", *IEEE Communications Surveys Tutorials*, vol. 21, no. 3, pp. 2671–2701, 2019. DOI: `10.1109/COMST.2019.2896380`.

[64] C. Kolias, G. Kambourakis, A. Stavrou, and S. Gritzalis, "Intrusion detection in 802.11 networks: Empirical evaluation of threats and a public dataset", *IEEE Communications Surveys Tutorials*, vol. 18, no. 1, pp. 184–208, 2016. DOI: `10.1109/COMST.2015.2402161`.

[65] T. T. Bhavani, M. K. Rao, and A. M. Reddy, "Network intrusion detection system using random forest and decision tree machine learning techniques", in *First International Conference on Sustainable Technologies for Computational Intelligence*, Springer, 2020, pp. 637–643.

[66] P. Raviteja, M. S. V. S. Devi, M. Gowri, M. V. S. Krishna, and P. Prabhakar, "Implementation of machine learning algorithms for detection of network intrusion",

[67] M. Zaman and C.-H. Lung, "Evaluation of machine learning techniques for network intrusion detection", in *NOMS 2018-2018 IEEE/IFIP Network Operations and Management Symposium*, IEEE, 2018, pp. 1–5.

[68] I. Dutt, S. Borah, I. K. Maitra, K. Bhowmik, A. Maity, and S. Das, "Real-time hybrid intrusion detection system using machine learning techniques", in *Advances in Communication, Devices and Networking*, Springer, 2018, pp. 885–894.

[69] Y. Zhou, G. Cheng, S. Jiang, and M. Dai, "An efficient intrusion detection system based on feature selection and ensemble classifier", *arXiv preprint arXiv:1904.01352*, 2019.

[70] D. Perez, M. A. Astor, D. P. Abreu, and E. Scalise, "Intrusion detection in computer networks using hybrid machine learning techniques", in *2017 XLIII Latin American Computer Conference (CLEI)*, IEEE, 2017, pp. 1–10.

[71] S. Shalev-Shwartz and S. Ben-David, *Understanding Machine Learning: From Theory to Algorithms*. Cambridge University Press, 2014. DOI: 10.1017/CBO9781107298019.

[72] L.-P. Chen, *Mehryar Mohri, Afshin Rostamizadeh, and Ameet Talwalkar: Foundations of machine learning, second edition*, 2nd ed. 2018.

[73] K. D. Foote, "A brief history of machine learning - dataversity", *Dataversity*, Mar. 2019. [Online]. Available: https://www.dataversity.net/a-brief-history-of-machine-learning/# (visited on 07/08/2020).

[74] R. Ashmore, R. Calinescu, and C. Paterson, "Assuring the machine learning lifecycle: Desiderata, methods, and challenges", *CoRR*, vol. abs/1905.04223, 2019. arXiv: 1905.04223. [Online]. Available: http://arxiv.org/abs/1905.04223.

[75] S. Kotsiantis, D. Kanellopoulos, and P. Pintelas, "Data preprocessing for supervised leaning", *World Academy of Science, Engineering and Technology, International Journal of Computer, Electrical, Automation, Control and Information Engineering*, vol. 1, pp. 4104–4109, 2007.

[76] S. Zhang, C. Zhang, and Q. Yang, "Data preparation for data mining.", *Applied Artificial Intelligence*, vol. 17, pp. 375–381, May 2003. DOI: 10.1080/713827180.

[77] J. Heaton, "An empirical analysis of feature engineering for predictive modeling", in *SoutheastCon 2016*, 2016, pp. 1–6. DOI: 10.1109/SECON.2016.7506650.

[78] U. Khurana, H. Samulowitz, and D. Turaga, "Feature engineering for predictive modeling using reinforcement learning", 2017. arXiv: 1709.07150 [cs.AI].

[79] "How to select algorithms for azure machine learning", *docs.microsoft.com*, Jul. 2020. [Online]. Available: https://docs.microsoft.com/en-us/azure/machine-learning/how-to-select-algorithms.

[80] D. Mladenić, "Feature selection for dimensionality reduction", in *Subspace, Latent Structure and Feature Selection*, C. Saunders, M. Grobelnik, S. Gunn, and J. Shawe-Taylor, Eds., Berlin, Heidelberg: Springer Berlin Heidelberg, 2006, pp. 84–102, ISBN: 978-3-540-34138-3.

[81] *International journal on computer science and engineering*, 3rd ed. Engg Journals Publications, 2011, pp. 1787–1797. [Online]. Available: http://www.enggjournals.com/ijcse/doc/IJCSE11-03-05-051.pdf.

[82] Y. Charfaoui, "Hands-on with feature selection techniques: Wrapper methods", *Medium*, Jan. 2020. [Online]. Available: https://heartbeat.fritz.ai/hands-on-with-feature-selection-techniques-wrapper-methods-5bb6d99b1274 (visited on 07/19/2020).

[83] ——, "Hands-on with feature selection techniques: Embedded methods", *Medium*, Jan. 2020. [Online]. Available: https://heartbeat.fritz.ai/hands-on-with-feature-selection-techniques-embedded-methods-84747e814dab (visited on 07/19/2020).

[84] F. Yang and K. Mao, "Robust feature selection for microarray data based on multicriterion fusion", *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, vol. 8, no. 4, pp. 1080–1092, 2010.

[85] J. Chu, I. Moon, and M. Mun, "A supervised feature projection for real-time multifunction myoelectric hand control", in *2006 International Conference of the IEEE Engineering in Medicine and Biology Society*, 2006, pp. 2417–2420. DOI: `10.1109/IEMBS.2006.259659`.

[86] M. Sewell, "Principal component analysis", 2007.

[87] I. Joliffe and B. Morgan, "Principal component analysis and exploratory factor analysis", *Statistical methods in medical research*, vol. 1, no. 1, pp. 69–95, 1992.

[88] L. I. Smith, "A tutorial on principal components analysis", Department of Computer Science, University of Otago, New Zealand, PO Box 56, Dunedin, Otago, New Zealand, Tech. Rep., Feb. 2002.

[89] J. C. Lam, K. K. Wan, K. Cheung, and L. Yang, "Principal component analysis of electricity use in office buildings", *Energy and buildings*, vol. 40, no. 5, pp. 828–836, 2008.

[90] N. Sánchez-Maroño, A. Alonso-Betanzos, and R. M. Calvo-Estévez, "A wrapper method for feature selection in multiple classes datasets", in *Bio-Inspired Systems: Computational and Ambient Intelligence*, J. Cabestany, F. Sandoval, A. Prieto, and J. M. Corchado, Eds., Berlin, Heidelberg: Springer Berlin Heidelberg, 2009, pp. 456–463, ISBN: 978-3-642-02478-8.

[91] Jianchang Mao, K. Mohiuddin, and A. K. Jain, "Parsimonious network design and feature selection through node pruning", in *Proceedings of the 12th IAPR International Conference on Pattern Recognition, Vol. 3 - Conference C: Signal Processing (Cat. No.94CH3440-5)*, vol. 2, 1994, 622–624 vol.2. DOI: `10.1109/ICPR.1994.577060`.

[92] C. Freeman, D. Kulić, and O. Basir, "Feature-selected tree-based classification", *IEEE Transactions on Cybernetics*, vol. 43, no. 6, pp. 1990–2004, Dec. 2013, ISSN: 2168-2275. DOI: `10.1109/TSMCB.2012.2237394`.

[93] P. Pudil, J. Novovičová, and J. Kittler, "Floating search methods in feature selection", *Pattern recognition letters*, vol. 15, no. 11, pp. 1119–1125, 1994.

[94] S. S. Khan and M. G. Madden, "One-class classification: Taxonomy of study and review of techniques", *The Knowledge Engineering Review*, vol. 29, no. 3, pp. 345–374, 2014.

[95] D. M. J. Tax, "One-class classification", PhD thesis, Delft University of Technology, 2001.

[96] M. M. Moya, M. W. Koch, and L. D. Hostetler, "One-class classifier networks for target recognition applications", *STIN*, vol. 93, p. 24 043, 1993.

[97] G. Ritter and M. T. Gallegos, "Outliers in statistical pattern recognition and an application to automatic chromosome classification", *Pattern Recognition Letters*, vol. 18, no. 6, pp. 525–539, 1997.

[98] C. M. Bishop, "Novelty detection and neural network validation", *IEE Proceedings-Vision, Image and Signal processing*, vol. 141, no. 4, pp. 217–222, 1994.

[99] P. Truong, "Approaches to anomaly detection", *Medium*, 2020. [Online]. Available: `https://medium.com/safetycultureengineering/approaches-to-anomaly-detection-20de4983d23` (visited on 09/01/2020).

[100] V. Nasteski, "An overview of the supervised machine learning methods", *HORIZONS.B*, vol. 4, pp. 51–62, Dec. 2017. DOI: `10.20544/HORIZONS.B.04.1.17.P05`.

[101] C. J. Burges, "A tutorial on support vector machines for pattern recognition", *Data mining and knowledge discovery*, vol. 2, no. 2, pp. 121–167, 1998.

[102] J. C. Platt, J. Shawe-Taylor, A. J. Smola, and R. C. Williamson, "Estimating the support of a high-dimensional distribution",

[103] M. Amer, M. Goldstein, and S. Abdennadher, "Enhancing one-class support vector machines for unsupervised anomaly detection", in *Proceedings of the ACM SIGKDD workshop on outlier detection and description*, 2013, pp. 8–15.

[104] Y. Wang, A. Miner, J. Wong, and P. Uppuluri, "Improving feature selection in anomaly intrusion detection using specifications", in *Distributed Computing and Internet Technology*, R. K. Ghosh and

H. Mohanty, Eds., Berlin, Heidelberg: Springer Berlin Heidelberg, 2005, pp. 468–468, ISBN: 978-3-540-30555-2.

[105] Y. Wang, J. Wong, and A. Miner, “Anomaly intrusion detection using one class svm”, in *Proceedings from the Fifth Annual IEEE SMC Information Assurance Workshop, 2004.*, IEEE, 2004, pp. 358–364.

[106] K.-L. Li, H.-K. Huang, S.-F. Tian, and W. Xu, “Improving one-class svm for anomaly detection”, in *Proceedings of the 2003 International Conference on Machine Learning and Cybernetics (IEEE Cat. No. 03EX693)*, IEEE, vol. 5, 2003, pp. 3077–3081.

[107] F. T. Liu, K. M. Ting, and Z.-H. Zhou, “Isolation forest”, in *2008 Eighth IEEE International Conference on Data Mining*, IEEE, 2008, pp. 413–422.

[108] W.-R. Chen, Y.-H. Yun, M. Wen, H.-M. Lu, Z.-M. Zhang, and Y.-Z. Liang, “Representative subset selection and outlier detection via isolation forest”, *Analytical Methods*, vol. 8, no. 39, pp. 7225–7231, 2016.

[109] M. M. Breunig, H.-P. Kriegel, R. T. Ng, and J. Sander, “Lof: Identifying density-based local outliers”, *SIGMOD Rec.*, vol. 29, no. 2, pp. 93–104, May 2000, ISSN: 0163-5808. DOI: `10.1145/335191.335388`. [Online]. Available: `https://doi.org/10.1145/335191.335388`.

[110] M. Alshawabkeh, B. Jang, and D. Kaeli, “Accelerating the local outlier factor algorithm on a gpu for intrusion detection systems”, in *Proceedings of the 3rd Workshop on General-Purpose Computation on Graphics Processing Units*, 2010, pp. 104–110.

[111] Y. Ma, H. Shi, H. Ma, and M. Wang, “Dynamic process monitoring using adaptive local outlier factor”, *Chemometrics and Intelligent Laboratory Systems*, vol. 127, pp. 89–101, 2013, ISSN: 0169-7439. DOI: `https://doi.org/10.1016/j.chemolab.2013.06.004`. [Online]. Available: `http://www.sciencedirect.com/science/article/pii/S0169743913001202`.

[112] A. Lazarevic, L. Ertöz, V. Kumar, A. Ozgur, and J. Srivastava, “A comparative study of anomaly detection schemes in network intrusion detection”, in *SDM*, 2003.

[113] E. Parzen, “On estimation of a probability density function and mode”, *The annals of mathematical statistics*, vol. 33, no. 3, pp. 1065–1076, 1962.

[114] M. P. Wand and M. C. Jones, “Multivariate plug-in bandwidth selection”, *Computational Statistics*, vol. 9, no. 2, pp. 97–116, 1994.

[115] D. Scott, *Multivariate density estimation: Theory, practice, and visualization: Second edition.* Mar. 2015, pp. 1–360, ISBN: 9780471697558. DOI: `10.1002/9781118575574`.

[116] L. Tarassenko, P. Hayton, N. Cerneaz, and M. Brady, “Novelty detection for the identification of masses in mammograms”, 1995.

[117] D. Reynolds, “Gaussian mixture models”, in *Encyclopedia of Biometrics*, 2009.

[118] M. Sauter, *From GSM To LTE-Advanced Pro And 5G*, 10-2017. John Wiley and Sons LTD, 2011, ISBN: 9781119346869.

[119] P. Martins, A. B. Reis, P. Salvador, and S. Sargento, “Physical layer anomaly detection mechanisms in iot networks”, in *NOMS 2020 - 2020 IEEE/IFIP Network Operations and Management Symposium*, 2020, pp. 1–9. DOI: `10.1109/NOMS47738.2020.9110323`.

[120] S. Sadowski and P. Spachos, “Rssi-based indoor localization with the internet of things”, *IEEE Access*, vol. 6, pp. 30 149–30 161, 2018.

[121] P. Romero and K. A. Dighe, “Fast and unsupervised classification of radio frequency data sets utilizing machine learning algorithms”, 2015.

[122] B. Faria, “Quais fatores influenciam na propagacao das ondas de radio?”, *Teletronix*, Nov. 2019. [Online]. Available: `https://teletronix.com.br/blog/quais-fatores-influenciam-na-propagacao-das-ondas-de-radio/` (visited on 09/11/2020).

[123] M. Dash and H. Liu, “Feature selection for classification”, *Intelligent data analysis*, vol. 1, no. 3, pp. 131–156, 1997.

[124] G. Chandrashekar and F. Sahin, “A survey on feature selection methods”, *Computers & Electrical Engineering*, vol. 40, no. 1, pp. 16–28, 2014.

[125] T.-T. Wong, “Performance evaluation of classification algorithms by k-fold and leave-one-out cross validation”, *Pattern Recognition*, vol. 48, no. 9, pp. 2839–2846, 2015. DOI: 10.1016/j.patcog.2015.03.009.

[126] A. Bhandari, “Standardization vs normalization”, *Analytics Vidhya*, Apr. 2020. [Online]. Available: https://www.analyticsvidhya.com/blog/2020/04/feature-scaling-machine-learning-normalization-standardization/ (visited on 09/12/2020).

[127] A. Zeiler, R. Faltermeier, I. Keck, A. Tomé, C. Puntonet, and E. Lang, “Empirical mode decomposition - an introduction”, Jul. 2010, pp. 1–8. DOI: 10.1109/IJCNN.2010.5596829.