Lightweight Data-Driven Anomaly Detection for IoT-Based Smart Grids: Capabilities and Limitations

Master's thesis in Computer Science and Engineering

Lisa Eskilson & Alexander Lager Carvalho

Department of Computer Science and Engineering
CHALMERS UNIVERSITY OF TECHNOLOGY
UNIVERSITY OF GOTHENBURG
Gothenburg, Sweden 2024

Master's thesis 2024

Lightweight Data-Driven Anomaly Detection for IoT-Based Smart Grids: Capabilities and Limitations

Lisa Eskilson
Alexander Lager Carvalho

UNIVERSITY OF GOTHENBURG

CHALMERS
UNIVERSITY OF TECHNOLOGY

Department of Computer Science and Engineering
Chalmers University of Technology
University of Gothenburg
Gothenburg, Sweden 2024

Lightweight Data-Driven Anomaly Detection for IoT-Based Smart Grids: Capabilities and Limitations

Lisa Eskilson
Alexander Lager Carvalho

Supervisor: Magnus Almgren, CSE
Advisor: Wissam Aoudi, Clavister AB
Examiner: Romaric Duvignau, CSE

Master's Thesis 2024
Department of Computer Science and Engineering
Chalmers University of Technology and University of Gothenburg
SE-412 96 Gothenburg
Telephone +46 31 772 1000

Typeset in LaTeX
Gothenburg, Sweden 2024

Lightweight Data-Driven Anomaly Detection for IoT-Based Smart Grids: Capabilities and Limitations

Lisa Eskilson
Alexander Lager Carvalho
Department of Computer Science and Engineering
Chalmers University of Technology and University of Gothenburg

Abstract

The integration of Internet of Things devices in critical infrastructure, such as the smart grid, has made it possible to monitor and manage energy distribution with increased efficiency. However, as these devices become more complex and interconnected, detecting physical tampering or data manipulation by malicious actors, such as the Sandworm attack in Ukraine, becomes increasingly challenging. One way that has shown promise in addressing this problem is the use of lightweight data-driven anomaly detection techniques.

In this thesis, PASAD and USAD, two state-of-the-art lightweight data-driven anomaly detection algorithms were selected and evaluated using a series of experiments simulating common attacks against smart grids as suggested by relevant research. These experiments aimed to investigate the viability of these algorithms in IoT-based smart grids. More specifically, the experiments include two different attack areas, namely OT- and network-level attacks, which were crafted by manipulating real smart grid operational data. These experiments were evaluated using time series-aware metrics to get a fair assessment of the efficacy of the algorithms.

The results from the experiments were used to evaluate the viability of lightweight data-driven anomaly detection algorithms and their capabilities and limitations were highlighted. Furthermore, the knowledge acquired from executing the experiments was used to propose guidelines for the development of an event management system that handles alerts produced by different models to provide valuable and actionable information to the OT operator.

The selected algorithms were successful in detecting various long-duration attacks with stealth characteristics, while other, shorter and more direct attacks, were significantly harder to detect. Despite this, these lightweight data-driven anomaly detection algorithms proved to be a good fit for the experiments evaluated in this thesis.

Keywords: Lightweight Anomaly Detection, Machine Learning, Internet of Things, Smart Grid

Acknowledgements

We would like to thank our supervisor Magnus Almgren for his support and guidance throughout this thesis. A special thanks to Wissam Aoudi, our supervisor at Clavister, for advising and helping us with anything as well as donating his time every week to check in. Lastly, we want to thank Clavister and CISSAN as well as iTrust, Centre for Research in Cyber Security, Singapore University of Technology and Design, for providing the resources needed for this thesis.

Lisa Eskilson & Alexander Lager Carvalho, Gothenburg, August 2024

Contents

List of Figures

List of Tables

1

Introduction

This chapter introduces the topic of this thesis with a brief description of The Internet of Things (IoT) and its relevance to the smart grid domain, highlighting how it enhances efficiency and optimisation by improving energy management and information flow. These advancements bring significant benefits but also introduce vulnerabilities that must be addressed to ensure security. Additionally, the chapter underscores the role of Intrusion Detection Systems (IDS) in mitigating these security concerns.

1.1 Internet of Things

IoT is a revolutionary concept, empowering cheap and energy-efficient devices to seamlessly exchange locally gathered data amongst themselves [29]. IoT devices have transformed the way we interact with physical devices in homes, vehicles and industry by enabling the collection and analysis of large amounts of data. This process has not only led to smart interactions between devices but also allowed for overall optimisations in various domains. An example of such an optimisation is smart energy management in buildings, where energy consumption can be reduced with the help of IoT-enabled sensors and thermostats [45].

Within the industrial sector, the IoT integration is particularly pronounced. This shift has reshaped industrial processes, enhancing efficiency and streamlining operations. By integrating sensors, actuators, and the ability to communicate between them, IoT devices enable real-time monitoring of industrial machinery. IoT devices not only enhance productivity and resource utilisation but also enable predictive maintenance and supply chain optimisation [29].

The smart grid is one of the largest potential IoT implementations and it relates to the combination of using IoT and power grids for connecting power producers and consumers. It solves some of the largest issues with conventional power grids such as information flow, energy waste, rising energy demand, efficiency, reliability and security [62].

The smart grid is an example of an Information Technology (IT) implementation in an Operational Technology (OT) environment. IT deals with data management and the flow of digital communication, while OT focuses on monitoring physical operations and the machinery utilised to execute them.

1.2 IoT cyber security challenges

While the IoT offers numerous advantages, its development has brought many cyber security risks [10]. IoT devices exhibit several weaknesses, such as weak or reused passwords, no account lockout mechanisms, poor encryption on devices when communicating, and lack of two-factor authentication [53]. These vulnerabilities make the nodes vulnerable and prone to being compromised by malicious actors through injection, or data manipulation. Intrusions into an IoT node can result in unauthorised access to the larger network, exposing vulnerabilities within the nodes. This leaves the nodes vulnerable to attacks to bypass more secure components of the system.

An example of an attack on IoT devices is the Mirai botnet, a Distributed Denial of Service (DDoS) attack, which utilised a botnet of IoT devices to target, among others, the DNS server provider Dyn [16]. The vulnerabilities that enabled this attack were weak passwords and outdated firmware in many devices, making them vulnerable to attacks.

An additional example underscoring the importance of cyber security in smart grids in particular is a recent cyber attack, called Sandworm which claimed to be arranged by Russia against Ukraine [55]. Employing a new OT-level technique known as Living off the Land (LotL), the attackers successfully compromised the victim's substation circuit breakers. This led to an unplanned power outage that coincided with a missile strike on important infrastructure throughout Ukraine. This attack highlights the urgent need for cyber security measures to secure OT systems.

Attacks like these are usually the result of not having enough security measures built into the devices as well as slow detection of incoming attacks. The vulnerability is due to the device's limited functions and its processing power. It is a challenge, as the devices must be cost-effective, and operate at low power to not make trade-offs on the security. This challenge is particularly pronounced in critical infrastructures such as the smart grid where the consequences may be great and can lead to compounded security risks. Here the abundance of OT systems requires different security measures than conventional IT security measures, which becomes evident when considering more complex attacks, such as the attack witnessed in Ukraine [55]. This is due to the fundamentally different operational environments and the critical nature of OT systems.

1.3 Intrusion detection systems

As IoT devices are becoming more complex and interconnected than ever it is becoming increasingly hard to detect physical tampering or data manipulation from malicious actors. In addressing this challenge, the integration of Intrusion Detection Systems (IDS) with Machine Learning (ML) has proven to be important, particularly in the realm of identifying and mitigating potential attacks, prior to any damage being inflicted [51].

One such solution is the use of signature-based IDS, also known as knowledge-based IDS, which utilises pattern-matching techniques to detect known attacks [34]. These techniques define anomalous behaviour and match intrusion signatures with those of previous intrusions in the database, raising an alarm when a match is found. An inherent limitation of signature-based IDS is its inability to detect zero-day attacks that have not been previously recorded.

Another type of IDS that has shown promise is the use of anomaly-based IDS. By establishing a baseline behaviour for nodes and networks, anomaly-based IDS can identify deviations from normal behaviour. However, the challenge with anomaly-based IDS lies in the non-trivial task of precisely defining the normal behaviour of nodes. This limitation can be addressed using ML [13]. By leveraging historical data and continuously monitoring data streams, ML can effectively define the normal behaviour of nodes, enabling the detection of behavioural changes attributable to malicious actions.

Smart grids, like other critical infrastructure systems, might be targeted by sophisticated attackers who may deploy undisclosed attack methods. Therefore, such environments benefit from using anomaly-based IDS, which is much more adaptive than signature-based IDS.

1.4 Problem motivation

In recent years, the frequency of attacks on OT environments has surged, underscoring the need for enhanced security solutions. This is reflected in the growing number of dedicated research reports, which conduct extensive investigations and comparisons to assess how Anomaly Detection (AD) systems perform in various contexts [22].

This thesis looks at three problems related to current research of AD systems in smart grid applications. Namely the scarcity of research on AD systems evaluated using smart grid datasets and especially lightweight implementations as well as the difficulty of handling events generated by such systems.

A crucial evaluation factor found in recent research reports is the choice of dataset, which influences the outcomes when evaluating different types of attacks on AD systems [22]. It is important to note that an algorithm's effectiveness in detecting a specific attack within one dataset, does not necessarily guarantee the same success in another. This key point is emphasised by Lamberts et al. [39], highlighting the substantial impact of dataset selection on the performance of the detection systems.

While there exists a lot of research on AD systems, there is a noticeable gap regarding their performance when applied to datasets from smart grids. Existing research often lacks in-depth insights into how these security systems specifically operate within the context of smart grids. This underscores the need for more focused and detailed exploration in this particular area.

Furthermore, this thesis aims to take existing research on algorithms for smart grids even further by specifically focusing on algorithms with lightweight characteristics. This aspect, discussed further in Section 1.6, is relevant to this thesis due to the need to limit the computational resources utilised by security measures in smart grids, as discussed in Section 3.1.

Finally, the results that are generated from IDS systems are not always easy to interpret and may differ between implementations. There exists little research on how to effectively handle events generated by the algorithms, due to the complexity of aggregating and generalising events. Therefore, this thesis will also look at how the events generated by multiple AD models can be processed in a way that it provides valuable information to the human operator which can have limited technical knowledge of IDS technology.

1.5 Thesis objectives

The research questions for this thesis encompass an exploration of smart grids, including specific requirements, as well as the evaluation of the algorithms within this domain. These provide a general idea of the areas that the results of this thesis aim to contribute to.

Research Questions

- How proficiently do state-of-the-art lightweight data-driven AD algorithms align with the requirements and characteristics of smart grid systems?
- How should the effectiveness of state-of-the-art lightweight data-driven AD algorithms applied in the smart grid sector be evaluated?
- How can the information from the AD algorithms be handled to produce valuable information for the human OT operator with limited knowledge of the inner workings of the algorithms?

The general research questions provide direction for the thesis and lay the foundation for the specific goals that this thesis aims to achieve.

Goals

1. Identify two or more state-of-the-art algorithms for comparison and evaluation. Investigate and choose appropriate evaluation metrics that accurately represent the performance of these algorithms.
2. Investigate and evaluate the potential strengths and limitations of various state-of-the-art AD systems designed for IoT devices in the smart grid by using data from the same domain. The investigation encompasses the analysis of identifiable attack types, as well as potential vulnerabilities that might not be detected by these systems.
3. Investigate how events generated by multiple anomaly-detection models should be handled, in a way that they provide valuable information to the human operator. Propose a theoretical solution to improve the state-of-the-art if

possible.

1.6 Thesis scope and limitations

As previously mentioned, research on AD algorithms is advancing at a rapid pace, with continuous evolution in the associated ML models. These models can in some cases become quite mathematically complex, warranting a survey in their own right. This thesis will not explore the composition of these ML models in a mathematical sense but rather treat them as black boxes. Instead, the focus lies on evaluating how effectively these lightweight models align with the requirements of the smart grid domain, described in Section 3.1, as well as their ability to detect incoming attacks.

The implementations of the selected AD algorithms might differ in terms of optimisation levels and maturity. This thesis uses the respective algorithms as they have been implemented, and any significant modifications to the implementations to improve performance are outside the scope of this work.

An additional constraint is related to the resource requirements of the AD algorithms. The term that is commonly used to describe an algorithm suitable for devices with limited resources is lightweight. However, there is no definitive definition specifying the exact resource requirements for an algorithm to qualify as lightweight. In this thesis, this aspect is be explored and instead only requires that the algorithm is specifically developed to be utilised in IoT devices. To investigate the lightweight properties, the algorithms are tested on a powerful IoT device, a choice that is motivated in Section 5.5.2.

Finally, this thesis aims at evaluating the performance of IDS algorithms in an offline setting. This means that the algorithms are exposed to the entire dataset as opposed to data streams in real-time which is the way that these algorithms would be used in reality. The complexity of developing an environment simulating components of a smart grid infrastructure lies beyond the scope of this thesis and is not explored.

1.7 Thesis outline

In Chapter 1, an introduction is provided on the topic of IoT, its associated challenges, and the role of IDS in enhancing the security of smart grids. Then, the objectives of the thesis are outlined, including the research questions and goals. Finally, the scope and limitations of the thesis are discussed.

In Chapter 2, the background of smart grids and anomaly-based IDS is reviewed. Furthermore, the current state of research and related works in anomaly-based IDS algorithms for detecting abnormal operations in smart grids is explored.

A more detailed theoretical overview which serves as a foundation to the work done in subsequent chapters is given in Chapter 3. It starts with providing more information on various attacks that smart grids commonly are subject to. Then, insights are given into state-of-the-art anomaly-based IDS algorithms and their composition.

In Chapter 4, a description of the process of selecting the algorithms for evaluation by specifying requirements, potential candidates and their alignment is provided.

The methodology of the experiments which evaluate the chosen algorithms is detailed in Chapter 5. Firstly, a detailed description of the processes used in selecting and pre-processing datasets for the experiments is provided. Secondly, a review of the evaluation setup and attack design for the experiments is presented. Lastly, an overview of the hardware on which the algorithms were tested and the metrics used to evaluate the results from the experiments.

In Chapter 6 an in depth description is provided of the individual experiments as well as the respective results. These results are later discussed in Chapter 7, where the viability of the selected algorithms as well as their capabilities and limitations are highlighted.

Current challenges and solution guidelines on how to handle events produced by the AD systems are discussed in Chapter 8. Finally, Chapter 9 provides suggestions for future work and summarises the key findings of this thesis.

2

Background & Related work

This chapter presents the relevant background to understanding the foundations of the smart grid infrastructure as well as anomaly-based IDS. Additionally, the current research in AD is explored in the context of smart grids and ICS along with its potential shortcomings, followed by the management of events generated by the detection algorithm.

2.1 Smart grids

Smart grids are modern electrical networks that utilise advanced technologies, including Cyber Physical Systems (CPS), to enhance electrical management and distribution [20]. These grids use two-way flow communication, facilitating real-time management between diverse nodes in the electrical grid. This allows improved control over electricity, enabling more efficient distribution and consumption management. A smart grid ensures a stable and reliable electricity network, providing the ability to integrate electricity from different power sources, like solar and wind, and reduce costs, among others.

The smart grid consists of various components, controlled and operated through Information and Communication Technology (ICT). This technological infrastructure enables energy companies to efficiently control and regulate energy demand.

A central component in the smart grid is the Supervisory Control and Data Acquisition (SCADA) system, described in further detail in Section 2.1.1, which serves as a central tool for gathering and monitoring data from different parts of the grid. This data collection allows operators to make decisions, assert control, and take action when anomalies arise.

2.1.1 SCADA

SCADA refers to a system of hardware and software components that enables industrial organisations to control processes and monitor data in real-time [64]. SCADA systems are widely used in different industries such as energy, water and wastewater treatment, manufacturing, etc.

In the centre of a SCADA system, a supervisory system is used to provide the user interface for the human operators to monitor and control industrial processes [70].

This system combines the Master Terminal Unit (MTU) and the Human Machine Interface (HMI). The MTU is responsible for gathering data, storing information and handling communication with other devices or systems. The HMI graphically presents operational information stored in the MTU to the human operators.

To supply the supervisory system with data in the control level of the SCADA architecture, Remote Terminal Units (RTUs) and Programmable Logic Controllers (PLCs) are used. These data acquisition and control units, monitor and control equipment such as sensors and actuators. This equipment, in turn, gathers data from physical processes in the system and performs physical changes to them. This level is called the field level [19]. This process is illustrated in Figure 2.1, where the sensors gather data from the physical processes in the system, which in turn are delivered to the RTUs/PLCs. These devices, either with the help of the SCADA master or on their own, can propagate actions to the actuators in the system, which make changes to the physical processes.

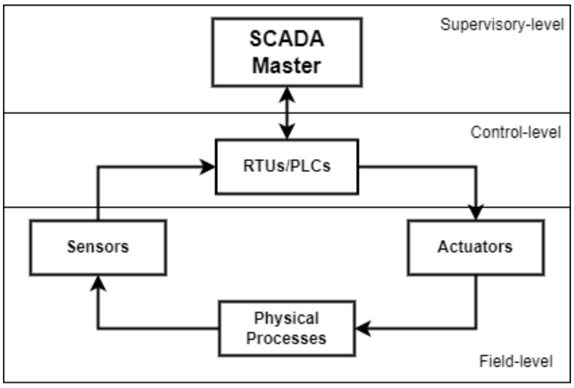

Figure 2.1: Scheme of the SCADA architecture illustrating parts of the supervisory-level, control-level and field-level.

The communication infrastructure within the SCADA system is essential for transmitting data between the supervisory system and the RTUs and PLCs within the substations. This data can be transmitted using different network protocols including Modbus, DNP3 and IEC 60870-5, among others [70]. In European countries, IEC 60870-5 is particularly prevalent and widely used as the preferred communication protocol in SCADA systems for the electrical industry [26].

2.1.2 Research on smart grid attacks

As discussed in Chapter 1, the smart grid, comprising networked technologies, may be vulnerable to cyber threats. Related research has investigated various vulnerabilities and threats to smart grids.

Significant advancements have been made in the security field, a topic that is discussed in the report by Samant et al. [61]. The report outlines the recent progress in research, focusing on common attacks targeting smart grids and the corresponding protective measures developed for various attacks. Notably, the report highlights the inclusion of IDS as a defence against malware and other threats. Additionally, it provides a comprehensive comparison of different technologies, discussing their respective advantages and disadvantages. For instance, ML can detect attacks in real-time but requires a lot of data to be effective which can pose a vulnerability in itself.

Y and Poornima [69] focus more on critical attacks, where they offer an overview of essential parts, requirements, and goals in cyber security attacks, alongside a review of potential cyber attacks on smart grids. They focus specifically on the critical attacks that have the potential to disrupt or damage power operations. The attacks discussed are the following: False Data Injection (FDI) attacks, Denial-of-Service (DoS), DDoS and spoofing attacks. Similar attacks are brought up by Stoytcheva et al. [66]. They provide a "defence in depth model" for organisations within the renewable energy systems area. They also provide an overview of different attacks, defence strategies, and models within the smart grid systems.

Relating to the concept of CIA introduced in Section 3.1, the integrity of data and an attack on it is considered to be an important threat to the smart grid. Liu et al. [44] exposed one such type of attack which was particularly effective against state estimation in electric power grids, namely FDI attacks. The attacks proved to allow manipulation of sensor measurements without being detected by existing algorithms. Due to the damage that can be inflicted while remaining undetected, this thesis heavily focuses on continuing research on evaluating AD algorithms on data with stealthy attacks.

2.2 Anomaly-based IDS background

Unlike their signature-based counterparts, which rely on predefined patterns of known attacks, anomaly-based IDSs operate by establishing a baseline of normal behaviour and identifying deviations from this baseline as potential attacks [60, 65]. The type of anomaly-based IDS that this thesis focuses on is data-driven techniques, particularly ML. These techniques encompass different types of ML, such as supervised and unsupervised learning.

2.2.1 Supervised learning

Supervised learning in IDS involves training models on labelled datasets where each data point can explicitly be marked as either normal or anomalous [63]. This enables the model to discern patterns within the data and learn the distinction between normal and abnormal behaviour. The trained model can then be used to predict the state of new data points and classify them based on the learned labels.

Ensuring effective generalisation is vital for the models, preventing both overfitting and underfitting on the training set. In the context of IDS, overfitting the model may lead to incorrect labelling of normal traffic as anomalous, and underfitting may fail to recognise unseen zero-day attacks as abnormal.

This type of learning requires human intervention to manually annotate the training set with the appropriate labels. Since this initial annotation is the foundation of the models, incorrect labels in the training set can lead the model to inaccurately distinguish between normal and anomalous behaviour.

2.2.2 Unsupervised learning

Unsupervised learning involves training a model on a dataset without an explicit classification label [63]. The goal of this type of learning technique is to discover inherent structures within the data, without any human intervention. The complex patterns discerned by the model enable the clustering of different data points based on the characteristics of their features.

In the context of IDS, unsupervised learning can be used to train the model exclusively on normal behaviour. During inference, the model can then flag abnormal entries that deviate from the learned behaviour. Common unsupervised techniques for AD include reconstruction-, clustering- and prediction-based methods [71].

Reconstruction-based AD involves a process of feature extraction or dimensionality reduction from raw data, which results in a more concise and representative set of features [5, 71]. During training, the model extracts information on historical patterns in the time-series data which are used in the inference phase where the features of a new observation are compared with the historical data. The greater the discrepancy, the higher the likelihood of being flagged as anomalous.

Clustering-based techniques project the data into a multidimensional space where the data points are grouped into clusters based on distance metrics [18, 71]. During training the data is projected and thresholds are set depending on the density and distance of data points in the cluster. In the inference stage, new observations are projected onto the multidimensional space and the distance to the normal cluster is used to classify the data point as normal or anomalous.

Prediction-based methods leverage historical time-series data to forecast the future state of the system accurately [71]. These models learn the pattern of data evolution over time and make a prediction at every time step, which is compared with the observed value.

2.3 Anomaly detection in smart grids

As the security concerns in smart grids are increasing so is the research on algorithms to protect them. As explained, research is being focused on both creating new lightweight AD algorithms as well as evaluating and comparing different types of algorithms in different environments.

2.3.1 Current state of research

Banik et al. [12] contribute to this discourse with an examination of AD approaches within the smart grid context. In total, twelve different types of implementation techniques are presented, offering a limited view of strengths and limitations, to identify research gaps for the development of a new AD algorithm for use in smart grids.

Lamberts et al. [40] offer further insights by evaluating industrial intrusion detection research publications in ICS, offering a visual representation of alerts produced by various algorithms when applied to the Secure Water Treatment (SWaT) dataset. The algorithms depicted include TABOR, Seq2Seq, PASAD, SIMPLE, Invariant, and Random. Furthermore, the same algorithms are also evaluated using point-based metrics as well as time-series aware metrics. This evaluation highlights the difficulty of comparing different algorithms based only on point-based metrics, as they fail to provide a coherent picture of the efficiency of the algorithms applied on time-series data. For this reason, this thesis focuses on using and implementing time-series aware metrics for evaluating the effectiveness of AD algorithms in dynamic environments such as smart grids.

Expanding the discussion, Mirzaee et al. [49] present a paper that covers an extensive literature review of the security of the smart grid, where they compare different studies on smart grids and which security area each one covers. Additionally, different attacks against smart grids are explained, along with the requisite countermeasures for each. Notably, the authors underscore the effectiveness of AD systems in mitigating DoS attacks.

Finally, Qi et al. [56], also explore different types of algorithms for AD in ICS. They examine and compare a range of popular semi-supervised AD algorithms, evaluating their performance on a power system dataset. The semi-supervised and supervised algorithms are compared, revealing that the semi-supervised approaches outperformed the supervised counterparts in detecting anomalies.

The papers discussed in this section underscore the importance of AD algorithms for secure smart grids. The papers provide valuable insights into the types of algorithms previously examined, particularly ICS, and the outcomes of such evaluations. They facilitate the exploration of relevant algorithms used in Chapter 4, as well as guide the prioritisation of testing for common attacks in Chapter 6.

2.3.2 Anomaly detection limitations in the smart grid

In literature regarding smart grid security, a common challenge is highlighted with using AD systems to detect incoming attacks or equipment malfunction, namely threshold definition. AD systems, particularly those based on residual-error analysis, often rely on predictive models to discern normal system behaviour from abnormal patterns [52]. The model predicts a future sensor measurement and uses a threshold to detect significant changes in the system. However, defining precise thresholds for flagging anomalies poses a considerable challenge, due to the inherent noise present in the data generated by diverse devices within a smart grid system [71]. Traditional threshold-based AD methods may struggle to adapt to the dynamic nature of smart grid data and may lead to high false positive rates or missed detections. Additionally, static thresholds may fail to capture subtle deviations indicative of emerging threats or equipment degradation.

Besides the threshold, there are other challenges present when dealing with the smart grid. Zhang et al. [71] describe some of the challenges that reduce the efficiency of AD in smart grids and can be condensed to these four points: (1) limited amount of data, (2) online, real-time detection, (3) stationarity, and (4) noisy inputs and disturbance.

Leveraging insights from prior research in smart grids and AD algorithms is essential for addressing highlighted challenges. Understanding these limitations is important for this thesis, as it provides valuable guidance on navigating future data management strategies and experiments. Incorporating insights from previous studies enables a more effective approach to addressing difficulties within smart grid environments.

2.4 Handling of alert events

With the transition from the conventional electrical grid to the smart grid, great amounts of data are generated every day by the IoT-devices in the grid. Consequently, there is a pressing need to effectively manage this data to fully harness the potential of this transition. This necessity has caused related research to delve into addressing these challenges.

Security Information and Event Management (SIEM) systems are central to event management, as they can detect, normalise and correlate security events [46]. A limitation of SIEMs is that they do not take into account the specific characteristics of the smart grid, which includes several legacy communication protocols. Radoglou-Grammatikis et al. [57] present SPEAR SIEM, which specifically targets the smart grid. Their architecture can be divided into three layers, (1) data capturing layer, (2) detection layer, and (3) correlation layer. In the correlation layer, the security events generated by the detection layer are correlated and a reputation value of each smart grid asset is calculated.

Zhang et al. [72] continue the trend of correlation analysis in intrusion behaviours. The authors propose a framework called Intrusion Action Based Correlation Framework (IACF), which aggregates security alerts, associates intrusion actions, extracts intrusion patterns and identifies different intrusion scenarios.

Albanese et al. [4] present a probability-based framework which analyses events that originate from both intrusion detection level as well as alert correlation processes. The authors found that current intrusion detection models and alert correlation models are ineffective at explaining a sequence of events identified in data streams. The proposed framework evaluates the probability that a sequence of events is not explainable by a given set of underlying models that observe the network.

These papers served as a foundation for the event management guidelines explored in Chapter 8.

3 Theory

This chapter provides the necessary theoretical foundation required to comprehend upcoming chapters. The first section mentions specific security requirements of the smart grid which are used extensively in this thesis. Furthermore, threats and vulnerabilities faced by smart grids are explored, explaining some of the most common attacks found in research and how they work. Following this, various AD algorithms and their functionalities are outlined. While this report does not delve deeply into how the algorithms work, the emphasis lies more on gaining an overview of their operations, strengths and weaknesses. Additionally, an overview of algorithm evaluation is provided, including common datasets and metrics utilised for IDS. Lastly, an explanation of two common protocols used in networks containing IoT devices is presented, offering a better understanding of upcoming experiments incorporating these protocols.

3.1 Security requirements of the smart grid

As delineated by the National Institute of Standards and Technology (NIST) there are three main requirements for maintaining information security in the smart grid, namely Confidentiality, Integrity and Availability [54].

- **Confidentiality**: Confidentiality refers to the protection of information from unauthorised access or disclosure. This includes protecting personal privacy and proprietary information.
- **Integrity**: A loss of integrity relates to the unauthorised modification of information, potentially harming decision-making regarding power management.
- **Availability**: In the Smart Grid, having timely and reliable access to information is crucial. A lack of availability can interrupt the ability to access data, which could negatively affect power delivery.

Historically, the most important requirement for power grids has been power availability with information integrity secondary. Confidentiality is regarded as the least critical requirement for power system reliability [54]. With the introduction of IoT to enable smart grids, confidentiality has become more important due to concerns related to the privacy of customer information.

More specifically, the requirements for the security measures used in a smart grid environment can be defined as in the report by Radoglou-Grammatikis and Sarigiannidis [58]. The requirements are more targeted to specific areas in the smart grid and some of them are described below:

- **Detecting a wide range of intrusions** - Malicious activity of different types and sources need to be detectable in the smart grid. This includes zero-day attacks and targeted attacks against specific hardware.
- **Timely intrusion detection** - Intrusions need to be detected within a reasonable amount of time.
- **High detection performance** - Various metrics such as Accuracy, Precision, True Positive Rate (TPR), False Positive Rate (FPR), True Negative Rate (TNR), and False Negative Rate (FNR) are calculated to evaluate classification performance. The balance of the different metrics is important to ensure that the IDS does not label abnormal data as normal.
- **Computational resources** - Several components in the smart grid, such as smart meters, are deployed on equipment that has constrained computational resources.
- **Scalability** - The smart grid consists of various components with different underlying technologies. Therefore, IDS systems need to be adaptable to be able to monitor a wide range of different components and technologies.
- **Friendly user interface** - The information generated by the IDS should be presented in such an efficient way that the human OT operator is able to quickly act on it.

These security requirements lie at the foundation of the security considerations of this thesis, such as algorithm selection in Section 4. These requirements also aid in the selection of evaluation metrics as the need for timely intrusion detection, high detection performance and minimising of computational resources is underscored.

3.2 Attacks on smart grids

Due to the complexity of components within a smart grid, numerous cyber attack scenarios are possible. These attacks have the potential to disrupt critical infrastructure, compromise sensitive data, and inflict physical damage. Among the most common threats, are FDI attacks, Man-in-the-middle attacks (MitM), LoTL, Targeted attacks, and DoS, as highlighted by relevant research in Section 2.1.2. To explain the outcome of these attacks, each attack is detailed below.

3.2.1 False Data Injection attacks

FDI attacks are malicious tactics where an attacker alters or inserts incorrect input data within a system to manipulate its output [27, 67]. By manipulating estimated values and introducing incorrect data, attackers aim to mislead decision-making processes, posing significant risks to critical components of the smart grid, for example, the SCADA centre. In such attacks, adversaries may target various system components, including IoT devices to exploit vulnerabilities. An example of such an attack involves an adversary manipulating data from a node within the system, for example, a sensor. This alteration can mislead the system into monitoring inaccurate values, leading it to make erroneous decisions based on the tampered data. The result of such attacks can lead to serious consequences, and threaten the integrity and functionality of the entire smart grid infrastructure.

3.2.2 Man-in-the-Middle attack

A MitM attack occurs when an attacker inserts themselves between the communication process between two parties, intercepting and potentially altering the data exchanged between them [36]. This intrusion allows the attacker to eavesdrop on sensitive information or manipulate the communication, while remaining undetected. By effectively becoming a silent intermediary, the attacker can inject malicious code into the IP packets or redirect packets to unintended IP addresses, all without the knowledge of the communicating parties.

3.2.3 Living off The Land attack

LoTL is an attack that involves an attacker carrying out stealthy actions without triggering alerts from security systems [68]. LoTL attacks, compared to conventional malware, operate without leaving a trace of code or scripts. Instead, the attacker uses pre-existing, legitimate tools already present in the target system. This method makes the attacks difficult to detect by security measures, often allowing them to persist within the victim's environment for a long time, ranging from weeks to years. In the Sandworm attack on the power system in Ukraine discussed in Section 1.2, hackers successfully infiltrated the OT environment and likely had access to the SCADA system for a duration of three months [32].

3.2.4 Targeted attacks

Targeted attacks typically involve singling out an organisation or a specific system as the primary target. The motivations behind such attacks can vary; the attacker may have a particular interest in the target or may have been contacted to launch an attack against it [50]. Targeted attacks are usually more harmful than non-targeted attacks because the attack is tailored to exploit vulnerabilities specific to the particular system or organisation. Therefore, relying solely on signature-based IDS proves insufficient to stop these clever attacks. Given that attackers often tailor and adapt their methods to infiltrate systems like smart grids, predicting and preempting such attacks become challenging.

3.2.5 Denial-of-Service

DoS attacks occur when a malicious actor disrupts the normal function of a device within a network, rendering it unable to receive packets or respond to IP requests [17]. This is typically achieved by overwhelming the device with an excessive number of requests until it becomes unresponsive. A DoS attack is typically executed from one single device, whereas DDoS attacks involve multiple devices to launch the attack.

3.3 Anomaly detection algorithms

This section provides a detailed view of different state-of-the-art AD algorithms and describes the features within the context of ICS. The presented algorithms in this section are among those that are most closely aligned to the requirements discussed in Chapter 4.

3.3.1 PASAD

PASAD is one example of a state-of-the-art AD algorithm, which monitors network- and process-level activity with the aim of detecting stealth attacks [8]. More specifically it works by analysing a discrete time series and for every point it calculates a departure score which is a metric that represents the quantitative deviation from normal operations. As such this algorithm is distinctive from other traditional AD algorithms which make predictions of future sensor measurements, while PASAD learns the normal behaviour of the time series and can detect small deviations. When the departure score is above a certain threshold there have likely been structural changes to a node which can be indicative of an attack.

PASAD is particularly effective at detecting changes in data with deterministic patterns such as IP traffic in machine-to-machine communication where the data from normal operation rarely deviates from the mean. Examples of data with these kinds of patterns are voltage, frequency and current measurements. This kind of data is likely to have noise which makes thresholds hard to set. However, the construction of PASAD is inherently noise resilient since it uses elements from Singular Spectral Analysis which is a tool for noisy time series of measurements that effectively separates the deterministic part from the chaotic part. Therefore, it allows the observation of overarching patterns in the deterministic part of the time series.

3.3.2 NSIBF

Neural System Identification and Bayesian Filtering (NSIBF) [21] uses a neural network architecture to comprehend the functioning of CPS within a state-space model. Thereafter, a Bayesian filtering algorithm is utilised to detect anomalies, providing continuous monitoring of prediction certainty over time.

This algorithm builds upon the concept of state estimation which is the process of using data from various sensors distributed throughout the network to estimate the current status or state of the entire system [2]. This iterative process continuously updates state models in response to new data, ensuring alignment with the dynamic properties of the system. The state estimator data processing schemes require three different pieces of information to compute the state of a system:

1. Measurements of system variables
2. Mathematical model of the system
3. Pseudo-measurements or prior information of system inputs and outputs.

The challenge with state estimation lies in the second point of building a mathematical model of the system, since in practice it is a step that requires lots of data from the different dynamics of a system. NSIBF uses a neural state identification scheme to build the mathematical model of the system, a process which builds on traditional state identification [31].

NSIBF is presented as an algorithm with capabilities of managing diverse complicated system behaviours, despite their complexity. Its primary strength lies in its ability to capture nonlinear and complex system dynamics effectively. Additionally, it provides robust protection against interference or sensor noise within the CPS.

3.3.3 MAD-GAN

MAD-GAN is a multivariate unsupervised AD algorithm that uses Generative Adversarial Networks (GANs) to capture complex multivariate correlations among multiple data streams [41]. It is designed with a framework that analyses all variables simultaneously to understand the hidden relationships or interactions between them, instead of treating each data stream separately and independently.

The GAN part of this algorithm handles time-series data. GAN is a prominent model in ML for generative AI [30]. The model consists of two neural networks, a generator and a discriminator and they operate through a competitive mechanism. The generator learns from a given training set to generate new data, similar to the original. Meanwhile, the discriminator provides indirect training, by assessing how realistic the input appears to be while dynamically updating itself. The generator attempts to trick the discriminator, facilitating unsupervised learning within the model.

In the MAD-GAN model, the generator and the discriminator are both constructed as Long-Short-Term Recurrent Neural Networks (LSTM-RNN). The generator generates a fake time series, while the discriminator attempts to distinguish between fake and real training data sequences. The trained generator and discriminator can then be utilised to identify anomalies using something called a Discrimination and Reconstruction Anomaly Score (DR-score).

One challenge with this method is the extended processing time required by LSTM-RNN to handle long sub-sequences. However, MAD-GAN has shown good outcomes when compared to alternative unsupervised AD methods.

3.3.4 USAD

USAD, published in 2020, is a noteworthy AD algorithm [11]. It operates as an unsupervised AD model tailored for multivariate time series based on trained autoencoders. Similar to MAD-GAN, in Section 3.3.3, the architecture of the autoencoder draws inspiration from GAN, featuring both an encoder and decoder structure. This design allows USAD to dynamically adapt and evolve in response to identified anomalies, enabling continuous improvement through unsupervised learning.

The algorithm consists of three different stages:

1. Data pre-processing involves normalising the data and splitting it into time windows of length k.
2. Training of the method, capturing the normal behaviour of the predetermined size of multivariate time series, resulting in the creation of an anomaly score for each time window.
3. The final step is AD where the models from step two are used. If the anomaly score is higher than the defined threshold, the corresponding time window is classified as abnormal.

The algorithm is evaluated on various datasets, including SWaT and WADI with good results compared to other state-of-the-art algorithms, particularly regarding the standard F1-score.

3.4 Algorithm evaluation

The evaluation of algorithms plays an important role in determining the efficacy and reliability of intrusion detection techniques. It is also a way of comparing different algorithms and finding strengths and weaknesses. This section provides an examination of the datasets and evaluation metrics commonly used to assess IDS algorithms in the dynamic environment of smart grids. Most of this section is based on the study by Lamberts et al. [40] which thoroughly researches the area of evaluations in industrial intrusion detection research.

3.4.1 Datasets

When evaluating data-driven algorithms, the choice of dataset is crucial as it serves as the foundational basis upon which performance metrics are calculated. These give insights into the efficacy in different domains.

Since smart grids and other critical infrastructure in the ICS domain generate vast amounts of sensitive data, the prospect of real-life data being openly shared in a public dataset is not very attractive due to privacy and security concerns. Therefore, most of the datasets that are evaluated in publications of IDS evaluations are private datasets. While private datasets allow researchers to evaluate algorithms using real-life data and therefore produce more accurate results, they hinder reproducibility and comparisons to related work. Hence, to strengthen any claims about a particular algorithm, public datasets can be used in conjunction with private datasets.

Morris-Power, SWaT and Electric Power and Intelligent Control (EPIC) are some of the most commonly used public datasets in evaluations of ICS environments [40].

The **Morris-Power** dataset includes Phasor Measurements Unit (PMU) or synchrophasor measurements and data logs from Snort [1]. These measurements are related to electric transmission system normal, disturbance, control and cyber attack behaviours. The labelled dataset consists of fifteen sets with thirty-seven power system event scenarios in each, which are divided into Natural Events, No Events, and Attack Events. The types of scenarios represented in the data are: Short-circuit fault, line maintenance, remote tripping command injection attack, relay setting change attack, and data injection attack.

The **SWaT** dataset, generated by the iTrust[1] SWaT testbed is a scaled-down replica of a real-world industrial water treatment plant, featuring a six-stage process [25, 48]. The dataset spans a continuous eleven-day period, operating twenty-four hours/day collection. Throughout the initial seven days, the data was gathered under normal operational conditions without any engineered attacks. However, during the remaining four days, various attack scenarios were launched.

Lastly, the **EPIC** dataset, developed by iTrust[1], serves as a valuable resource for research and development in the domain of cyber security for power systems [3]. Similarly to SWaT, the EPIC testbed is a replica of a real-world power grid which generates data that reflects real-world scenarios in power grid operations. The dataset encompasses various components such as generators, transformers, PLCs, HMIs and SCADA systems.

3.4.2 Metrics

For comparing IDS systems it is useful to have different metrics that evaluate various aspects of performance. For this thesis two different types of metrics are relevant. The group of metrics evaluate the algorithm's efficacy in distinguishing between normal and abnormal operations. These can be broken down into the following types of metrics: confusion matrix, point-based metrics and time series-aware metrics. The second group of metrics evaluate how much resources are required for the IDS system to operate. These are more specific towards IoT use cases, where it is useful to keep the overhead of the IDS systems as low as possible and resource efficiency is instrumental.

A **confusion matrix** is a tool that is used to evaluate the performance of an algorithm using a set of classification metrics [24]. After the training phase of the algorithm, for every data point in the dataset, the classification label (alarm or no alarm) is compared with the output of the algorithm. By analysing the difference it is possible to build a matrix of the following metrics: True-Positive (TP), True-Negative (TN), False-Positive (FP), and False-Negative (FN).

Using different combinations of the four metrics from the confusion matrix, various **point-based** metrics can be derived. These express the performance of the system using a single value score and look at different aspects of the system, such as accuracy, precision, recall, miss-rate, specificity, fall-out and F1-score. Notably, these

[1] Dataset from iTrust, Centre for Research in Cyber Security, Singapore University of Technology and Design, available at `https://itrust.sutd.edu.sg/itrust-labs_datasets/dataset_info/`

metrics look at every entry in the dataset in isolation, computing an independent score for each.

In complex systems such as the smart grid, the data generated is inherently time-dependent where the current state of the system is heavily dependent on the previous state. Therefore, point-based metrics do not accurately reflect the performance of the IDS when dataset entries are analysed in isolation. Instead, **time series-aware** metrics are specifically designed to assess the performance of systems that use time-series data, by taking the temporal aspect of the data as well as the dynamic nature of the system into account. Such metrics include, detected scenarios, detection delay, enhanced Time-aware Precision and Recall (eTaPR) and affiliation.

eTaPR are time series-aware variants for traditional point-based metrics, namely, precision, recall, and F1 [28]. The enhanced Time-aware Recall (eTaR) updates the conventional recall metric, enhanced Time-aware Precision (eTaP) modifies the classic precision metric, and the new Time-aware F1 (eTaF1) combines these adjusted metrics to replace the traditional F1 score. These new metrics address known issues when adopting point-based metrics for time series analysis. Specifically, they consider an anomaly as a range rather than a single point, acknowledging that an anomaly can be detected even if only part of it is identified by the detection method.

Besides the metrics that evaluate the detection performance, other metrics that analyse resource utilisation are quite relevant. Given the multitude of components with IoT characteristics within the smart grid, it is important for algorithms to have minimal impact on the device. Therefore, metrics such as CPU/GPU usage, RAM utilisation, and model size are essential for assessing the suitability of different algorithms for deployment in devices with limited hardware resources.

3.5 Network protocols

In a network, various protocols facilitate different types of communication between devices. Among these protocols are Message Queuing Telemetry Transport (MQTT) and IEC 60870-5, both commonly utilised in smart grids. In this thesis, data from both of these protocols are analysed and manipulated to identify vulnerabilities specific to each protocol.

3.5.1 MQTT

The MQTT protocol is a data exchange protocol between IoT devices. It is specifically designed to enable asynchronous real-time communication among IoT devices operating under constraints such as limited bandwidth and resources [47]. It is run over TCP/IP and operates on a publish-subscribe model and allows devices to publish messages with specific topics, which other devices can then subscribe to receive relevant messages. For instance, a sensor measuring voltage can publish messages with the topic “volts”, allowing a monitoring unit to subscribe to this topic to receive updates on voltage levels. A device can have several subscribers and therefore enables one-to-many communication.

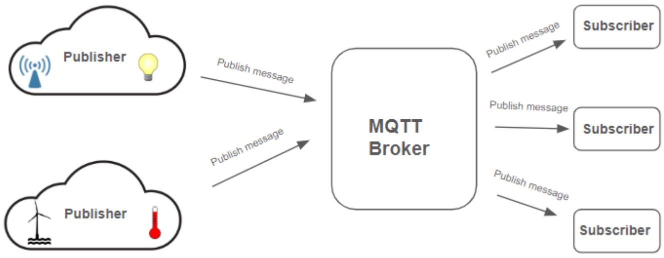

Figure 3.1: Diagram illustrating the communication process in the MQTT protocol. The publisher transmits the published messages, which the MQTT broker then forwards to subscribers.

The MQTT protocol consists of two fundamental components, an MQTT client and an MQTT broker. In a smart grid network, a client represents a device connected to the network, for example, a sensor. These clients do not communicate directly with each other, instead, their messages are intermediated by the broker. The broker handles the message routing and manages which clients subscribe to which topics. This communication process is presented in Figure 3.1. Additionally, the broker ensures that packets not delivered due to a connection disruption are correctly transmitted to the respective client once the connection is re-established. There are different types of brokers, but since the protocol is designed for a large-scale network, a cloud-based broker is usually used.

3.5.2 IEC 60870-5

IEC 60870-5, mentioned briefly in Section 2.1.1, is a collection of standards initially published in 1995 by the International Electrotechnical Commission (IEC) [14]. It specifies a set of telecontrol protocols with the aim of providing an open standard for the transmission of SCADA telemetry control and information for application in electrical industries. More specifically it is used for enabling communication between power grid substations and the central control center.

IEC 60870-5 encompasses several profiles, each specifying different protocols and communication profiles tailored for specific applications and network configurations.

- **IEC 60870-5-101** defines a standard for telecontrol tasks, specifying a communication protocol for serial transmission.
- **IEC 60870-5-102** defines the Open Systems Interconnection model-based communication.
- **IEC 60870-5-103** defines serial point-to-point communication.
- **IEC 60870-5-104** defines a communication profile for TCP/IP-based communication networks.

In the application layer of IEC 60870-5-104, the Application Protocol Data Unit

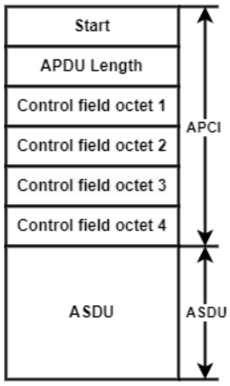

Figure 3.2: Structure of the APDU.

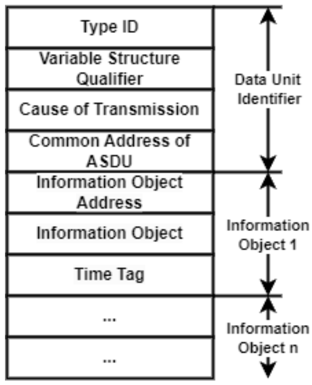

Figure 3.3: Structure of the ASDU.

(APDU) is the unit of information that encapsulates the data exchanged between the SCADA master and the RTUs. The APDU in turn consists of two parts, as illustrated in Figure 3.2, the Application Service Data Unit (ASDU) and Application Protocol Control Information (APCI). The APCI provides the control information necessary for the proper interpretation and handling of the data in the ASDU.

The ASDU is the data structure used to package system information such as measurements, control commands and other pertinent monitoring data. The structure of the ASDU, illustrated in Figure 3.3 consists of two main sections, the Data Unit Identifier which contains information about the data and the data itself inside the Information Object.

4

Anomaly detection algorithm selection

This chapter details the process of selecting algorithms for evaluation in the experimental part of the thesis. Here the requirements and possible candidates are specified, and then systematically evaluated.

4.1 Requirements

There is a wide range of algorithms with the same goal of preventing attacks on smart grids. Before the selection of algorithms for the comparisons in this thesis, a literature review was made. Each paper and algorithm was examined on the following requirements detailed below.

4.1.1 Inherent algorithm requirements

The requirements that pertain to the inherent features and architecture of the algorithm are described below.

Lightweight

A lightweight algorithm is designed to be efficient in memory requirements and computational resources. It is specifically designed to operate on devices with limited computational power, such as IoT devices, making them an essential requirement for smart grids. As mentioned in Section 1.6 there is no definitive definition of what resource requirements classify an algorithm as lightweight. The definition that this thesis uses for the term lightweight is an algorithm that specifically targets IoT devices.

Data-driven

A data-driven algorithm learns from and is trained on data to identify patterns, behaviours and relationships. It relies on large datasets to extract insights and make decisions based on the patterns it learns. By analysing the data it is trained on, the algorithm can learn from its own mistakes and adapt and improve its performance over time.

Multivariate

A multivariate algorithm is able to extract patterns from multiple features in a dataset. By considering multiple variables simultaneously, algorithms can find complex relationships and patterns that might otherwise be hidden when variables are analysed in isolation. This in turn allows for more accurate learning of complex system behaviour.

Time series

An algorithm capable of analysing time-series data can identify patterns in value changes over time. Time-series data is a sequence of data points that are collected at successive discrete time intervals, which allows for analysis of how variables change over time. The data generated by devices in a smart grid has this time-series structure and therefore it is crucial that the algorithm is able to analyse and extract meaningful patterns from it.

4.1.2 General requirements

Besides the inherent requirements of the algorithm, other more general requirements related to the implementation of the algorithm are described below.

Code availability

Access to the source code of a proposed algorithm is necessary as this thesis executes domain-specific experiments on the algorithms that test different strengths and weaknesses. Hence, the algorithm must either be open-source or permission to access the code must have been obtained from the authors.

Official implementation

Official implementation describes the instances where the authors of the paper provide the source code that was used to generate the results shown in the paper. However, sometimes a reference to this source code is omitted and instead third parties implement the algorithm based on the description in the paper.

Working code

If code is available, it should be functional in an experimental environment, able to compile and run without errors that alter the expected output. Furthermore, it should be executed within a reasonable time frame.

Documentation

Along with the available code, there must be documentation available, containing clear instructions outlining its functionality and execution process.

Versatility

The versatility of the algorithm implementation captures the ability of the implementation to adapt to different datasets as well as different types of data, such as process- and network-level data. Since several different datasets are used in this thesis, with a wide variety of features it is important to be able to easily run the algorithm with a new dataset without having to modify the implementation.

4.2 Candidates

The first step of identifying relevant candidates was through a thorough literature review. This included a review of relevant surveys that listed multiple state-of-the-art algorithms. These papers and others were examined with the inherent algorithm requirements, specified in Section 4.1.1, and the findings were documented. After conducting the literature review and exploring potential candidates, the following relevant algorithms were found.

The **Sequence-to-Sequence (Seq2Seq)** neural network algorithm, published by Kim et al. [35], in 2019, is designed for training and predicting ICS operational data, and handling and clarifying time-series attributes. This algorithm, which uses unsupervised training was evaluated on the SWaT dataset. The result in the report shows that the algorithm successfully detected 29 out of 36 attacks.

1D CNN, UAE, VAE, PCA, published by Kravchik and Shabtai [38], in 2022, suggests an algorithm utilising 1D Convolutional Neural Networks (CNNs), Variational Autoencoders (VAEs), shallow Undercomplete Autoencoders (UAEs), and Principal component analysis (PCA) for detecting physical-level ICS data. The algorithm was trained on both simulated and real datasets, including SWaT. The method proves to be good at detecting adversarial evasion attacks, however, it needs more testing for confirmation of credibility.

Time Automata and Bayesian netwORk (TABOR), published by Lin et al. [42], in 2018, is a graphical model-based design. The algorithm learns from observations of normal behaviour recorded by sensors within an ICS, enabling the detection of anomalies. It was trained on the SWaT dataset with good results in both run-time and precision.

PASAD is a lightweight data-driven algorithm, published by Aoudi et al. [8], in 2018. It is a novel approach aimed at detecting stealth attacks by constantly monitoring real-time sensor data to identify changes in process behaviour. PASAD has proven effective on microcontrollers, demonstrating its lightweight characteristics [9]. It was evaluated on three scenarios, including the SWaT testbed and network traffic originating from a water distribution plant. The result shows, that PASAD can be applied in real-world scenarios.

MAD-GAN is an unsupervised multivariate AD algorithm based on GAN, which was published by Li et al. [41] in 2019. This algorithm learns in an unsupervised manner to detect anomalies in real data. MAD-GAN was evaluated on the SWaT dataset, demonstrating good performance compared to other unsupervised detection algorithms.

VAE-LSTM is a hybrid unsupervised AD algorithm, published by Lin et al. [43], in 2020. The algorithm focuses on AD within a time series, which includes two components. The first component is the VAE, used to identify structured patterns in the time series. The second component is the LSTM-RNN, designed to capture the trend over a long duration. The algorithm is evaluated on five different time-series datasets, such as a time series tracking the number of taxi requests in NYC.

The **Lightweight Variational Auto-Encoder for Long-Term Memory (LW-LSTM-VAE)** algorithm, published by Fährmann et al. [23], in 2022, offers an unsupervised deep learning approach for AD in ICSs. The algorithm was evaluated on the SWaT and demonstrates good performance in AD efficiency. Notably, its lightweight architecture demands fewer parameters compared to other methods.

Neural System Identification and Bayesian Filtering (NSIBF), published by Feng and Tian [21], in 2021, is an AD method designed for time-series data. Utilising a unique neural network architecture, NSIBF effectively captures the dynamics of CPS within a state-space model. Additionally, it incorporates a Bayesian filtering model to monitor the uncertainty associated with the system's hidden state over time. NSIBF performance was evaluated on the SWaT dataset, showing good AD performance compared to other state-of-the-art algorithms.

UnSupervised Anomaly Detection for multivariate time series (USAD), published by Audibert et al. [11], in 2020, is a method based on autoencoder architecture, with learning inspired by GANs. With the help of this architecture, the algorithm can learn in an unsupervised way and thus isolate anomalies. USAD was evaluated on several datasets, including SWaT, with good results compared to other state-of-the-art algorithms in both anomaly detection and fast testing time.

4.3 Evaluation and selection

The process of evaluating the algorithms comprises two parts, the evaluation of inherent characteristics of the algorithm followed by the implementation of the algorithm.

4.3.1 Inherent algorithm characteristics

The papers identified in Section 4.2 were initially analysed and assessed according to the requirements delineated in Section 4.1.1. These requirements reflect on the inherent characteristics of the algorithms described in the papers, allowing the refinement of the list of potential algorithms for further consideration.

To classify the papers using the specified requirements some liberties had to be taken, more specifically for the lightweight and multivariate qualities of an algorithm. Whereas some papers specifically mentioned lightweight or multivariate, others did so more subtly while others not at all. As described in the requirement definition in Section 4.1, a lightweight algorithm is specifically developed for an IoT device, and only papers that explicitly described this characteristic, fulfilled this category. Similarly, for an algorithm to be classified as multivariate, it must explicitly mention it in relative terms, as otherwise it was assumed to be univariate.

As the results of the analysis in Table 4.1 indicate, most algorithms closely align with theoretical expectations. Only the Seq2Seq [35] algorithm failed to fulfil two requirements, namely lightweight and multivariate. The former proved to be the most challenging, with only five out of nine algorithms fulfilling it. This outcome was predictable, given the relatively lesser research in this area.

Inherent requirements algorithm evaluation				
Name	Lightweight	Multivariate	Data-Driven	Time Series
Seq2Seq [35]	No	No	Yes	Yes
1D CNN [38]	Yes	Yes	Yes	Yes
TABOR [42]	No	Yes	Yes	Yes
PASAD [6, 8, 9]	Yes	Yes	Yes	Yes
MAD-GAN [41]	No	Yes	Yes	Yes
VAE-LSTM [43]	No	Yes	Yes	Yes
LW-LSTM-VAE [23]	Yes	Yes	Yes	Yes
NSIBF [21]	Yes	Yes	Yes	Yes
USAD [11]	Yes	Yes	Yes	Yes

Table 4.1: Table for evaluation of algorithms based on the theoretical algorithmic properties explained in Section 4.1.1.

4.3.2 Algorithm implementation

Besides the inherent requirements that the selected algorithm should have, there were other important requirements to consider. These reflect more general requirements as related to the actual implementation of the algorithm as outlined in Section 4.1.2.

Primarily, access to the source code or compiled libraries was needed, either by gaining access privately or finding a publicly available implementation. While some papers provided access to the implementation used to get the results from the paper, others omitted it completely. Besides official implementations, some algorithms had unofficial implementations where the guarantees of replicating the results reported in the respective papers are likely low. As the resulting Table 4.2 shows, among the algorithms where access was obtained, only USAD was an unofficial implementation. However, the developer was assisted by some of the original authors and even provided with some code foundations used by the original authors.

After the availability of the implementation was evaluated, the algorithms were tested locally with the dataset accompanied in the source code. This stage was not about replicating the results in the paper, but rather analysing the functionality of the implementations as well as analysing how well documented the functions were. The result is reflected in Table 4.2 where only the PASAD, NSIBF and USAD implementations provided a meaningful result without extensive troubleshooting.

Finally, the versatility and adaptability of the functioning implementations were examined by exposing the implementations to different types of data using the SWaT dataset which includes both process-level data as well as network-level data. This aspect is very important because the more versatile an implementation is, the more efficient it becomes in testing different datasets with various types. Notably, the NSIBF implementation proved to not be suitable for detecting anomalies in network-level data as reflected in Table 4.2.

General requirements algorithm evaluation					
Name	Availability	Official	Working code	Docs	Versatility
Seq2Seq [35]	Yes[1]	Yes	No	1	—
1D CNN [38]	No	—	—	—	—
TABOR [42]	No	—	—	—	—
PASAD [6, 8, 9]	Yes*	Yes	Yes	5	5
MAD-GAN [41]	Yes[2]	Yes	No	2	—
VAE-LSTM [43]	Yes[3]	Yes	No	2	—
LW-LSTM-VAE [23]	No	—	—	—	—
NSIBF [21]	Yes[4]	Yes	Yes	2	2
USAD [11]	Yes[5]	No**	Yes	2	5

* The latest version of the private implementation of PASAD was provided by the original authors.
** The implementation[7] of USAD was made in close contact with the original authors.

Table 4.2: Table for evaluation of the implementation of the algorithms based on the properties explained in Section 4.1.2. The table utilises a dual rating system comprising both binary evaluations and a numerical scale ranging from 1 to 5.

As a result, the two algorithms which proved most suitable for this thesis were PASAD and USAD. The following outlines the reasons and motivations behind the selection of each algorithm:

PASAD:

- A lightweight algorithm that is very suitable for IoT devices.
- Shows good performance compared to other state-of-the-art algorithms.
- Simple to use and adapt for different dataset formats.

USAD:

- Shows good performance compared to other state-of-the-art algorithms.
- The code works without any adaptions.
- Easy to understand and use.

The algorithms selected in this chapter were used in the experiments outlined and performed in the upcoming chapters, where further detail is presented on specific implementation parameters such as window size or lag and training set size. These experiments evaluated both PASAD and USAD to determine their suitability for IoT devices in smart grids.

[1] `https://github.com/jukworks/swat-seq2seq`
[2] `https://github.com/LiDan456/MAD-GANs`
[3] `https://github.com/lin-shuyu/VAE-LSTM-for-anomaly-detection`
[4] `https://github.com/cfeng783/NSIBF`
[5] `https://github.com/manigalati/usad`

5
Experiment methodology

This chapter provides an overview of the methodology employed in this thesis. Firstly, it encompasses a review of the processes involved in selecting the datasets and how they were pre-processed. Secondly, it delves into the methodology employed for evaluating the algorithms, including the creation and simulation of the attacks. Thirdly, it outlines the method used for testing the algorithms on the hardware, to support the claim of their efficacy for deployment on IoT devices. Lastly, the metrics utilised to evaluate the obtained results are described.

5.1 Chosen datasets

This thesis utilises two different datasets. The first dataset, SWaT, as described in Section 3.4.1, is a public dataset that was used to demonstrate that the selected algorithms work as expected. SWaT is a well-established dataset widely used in many relevant papers, offering a reliable benchmark for confirming the expected performance before transitioning to evaluation with the second dataset, CISSAN. This dataset is specifically utilised for experimental purposes in this thesis. It comprises data sourced from power grids provided by Affärsverken. A more detailed description of both datasets will be presented in the following sections.

5.1.1 SWaT

The SWaT dataset contains two types of data, physical and network data [25]. Physical or OT data originates from sensors and actuators within the testbed and is collected in operational mode. The network data is obtained when the testbed transitions into operational mode. Both datasets are stored in separate CSV files, each labelled according to the type of data it represents.

The designed attacks encompass different types, targeting specific points within a CPS or multiple parts of the system. In total, thirty-six attacks were executed during the data collection period. The attacks vary in duration based on their type, while some have immediate effects, others require time to show.

As mentioned earlier, the SWaT dataset is utilised to validate the proper functioning and performance of the algorithms. Both PASAD and USAD have undergone prior evaluation using SWaT, establishing a foundation for comparison with previous results. This comparison provides a basis for assessing CISSAN, which has not been previously tested with these algorithms.

5.1.2 CISSAN

This thesis was made in collaboration with an industrial partner Clavister AB, which is part of the European research project CISSAN. As a result, Clavister AB provided real smart grid operational data including process- and network-level data collected by Affärsverken from power stations in Sweden [15].

CISSAN is a project focused on implementing and researching algorithms aimed at securing IoT devices in power grids. The primary focus lies in algorithms utilising ML, which underscores the emphasis on such algorithms within this thesis.

The data originates from three different substations in a power grid, Station 1, Station 2 and Station 3. The data consists of fourteen consecutive days of irregularly sampled IPv4 packet captures from an RTU in each of the substations, which contain sensitive OT information. This specific type of data, in the form of sensor measurements of physical processes within the substations, is communicated with the SCADA master and other IoT devices using the IEC 60870-5-104 and MQTT protocols respectively.

5.2 Data pre-processing

To ensure the algorithms ran effectively, it was essential that the data they were trained on was in the correct format and included the necessary parameters for evaluation. Before running the experiments, the datasets were pre-processed and adapted to suit the requirements of the respective algorithm.

5.2.1 Format

The CISSAN dataset is a collection of packet captures stored in a PCAPNG format. By applying specific filters in Wireshark, it was possible to sort the packets on the protocols that were going to be analysed in a specific experiment. The OT dataset was filtered based on either MQTT or IEC 60870-5 ASDU. This filtering was based on their frequent use in smart grids and relevance to the research topic, and it is also where some of the most sensitive data resides.

The dataset from the three different stations was then extracted into four distinct formats: IP headers of all the TCP traffic and OT data for IEC 60870-5 ASDU and MQTT. The formats were determined based on the relevance of the data and the features associated with each type. This enables the examination of different aspects of the network data flow.

The exported data contained a mixture of different types, including numeric values, addresses and strings. ML algorithms require input data to be numeric, as they rely on mathematical operations to learn patterns and make accurate predictions. Therefore, it was necessary to pre-process the data to ensure that all values were floating point numbers before applying the algorithms.

5.2.2 Feature engineering

Feature engineering is a critical step when dealing with ML algorithms as it has an impact on performance, accuracy and the generalisation ability of the model. By providing the model with the most relevant data and omitting non-deterministic data, the model only discerns patterns in the data of most importance within the system.

Due to the nature of how the packets were captured in a non-regular manner, the capture time was not taken into account as a feature for any of the CISSAN datasets. The time would introduce a source of randomness which could lead to worse results as the integration of non-deterministic data into a data-driven algorithm could potentially influence the outcome.

The irregular sampling introduced noise to the sensor values contained in the MQTT and SCADA protocols, negatively affecting the results. The impact of noise was reduced by using statistical features. Statistical features are commonly employed in ML to mitigate noise in datasets by providing a summary of the underlying data distribution. Metrics such as median, mean, standard deviation, and variance are less vulnerable to extreme data points compared to individual data points. By summarising the data distribution, statistical features use statistical functions to mitigate the influence of extreme data points, which are often the source of significant noise. This helped to improve the quality of input data which in turn helped to improve the AD.

The selection of features for the IP header data was to include as many IPv4 headers as possible while excluding ones containing randomness. Two features excluded were checksum and fragment offset, as they can be arbitrary and hold no relevance for future experiments.

The SCADA protocol IEC 60870-5-104 outlined in Section 3.5.2 contains an ASDU where the data resides. The most pertinent features of the ASDU are the common address, Information Object Address (IOA) and the information object. The common address is the ASDU address of the substation where the data unit originates. The IOA and information object are the identifier of the type of data or the sensor identification as well as the value itself. These features are the most meaningful and non-deterministic, which aid the algorithms in finding meaningful patterns.

The selection of features from the MQTT protocol, as outlined in Section 3.5.1, was based on their relevance to the experimental objectives. Among these features, the most important ones were the message id and the message value. The message value, which contains the measurement value sent in an MQTT publish message, along with the message id, describes the type of data that has been sent and where in the system it came from.

5.3 Evaluation setup

The AD algorithms underwent initial training using data without attacks. Then, the algorithms were tested on datasets containing manipulated data. As indicated in Section 5.1.1, before the introduction of the CISSAN data to the algorithms, the algorithms were evaluated using the SWaT dataset, to gain an insight into how the algorithm implementations perform compared to the respective papers.

5.3.1 SWaT evaluation

Before proceeding with the evaluation of USAD and PASAD using data from smart grids, both algorithms were tested on the established SWaT dataset, as detailed in Appendix A.2.

This evaluation aimed to verify that the algorithms functioned as intended, consistent with the specifications outlined in their respective papers. By using the SWaT dataset, which had previously been utilised to assess both algorithms, this process established a benchmark for their performance, with results available through various types of reports.

5.3.2 Algorithm parameters

Several parameters can be tuned for the algorithms. This thesis only focuses on three parameters that are common for the general algorithms and do not require modifying the inner workings of the algorithm, only how it interacts with data.

The first parameter that is important in any ML model is the size of the training set. As both models were trained using data without attacks and evaluated using attack data, the sample must be sufficiently large for the models to learn complex patterns in the system. However, as the size of the training set increases so does the training time, as well as the amount of data needed, which is usually limited. The size of the training set varies in different experiments due to filtering data on various features, depending on the type of experiment conducted. For instance, an experiment may exclusively utilise data containing TTL values in the IP header.

The other parameter that was considered is the lag in PASAD or the window size in USAD. This parameter is fundamental to how time-series AD algorithms work since the data points are not evaluated in isolation, but in sequence. The lag/window size simply defines the number of data points that should be considered sequentially. By increasing the lag/window size, the model considers more historical data points and can find longer-term patterns in the data. Since more data points are evaluated at every time point the computational cost and training time increases as well. On the other hand, by decreasing the lag/window size, the model can detect short-term fluctuations in the data and also decrease the training time.

The final parameter that needed to be set was the departure/anomaly score threshold, which represents the lowest score that results in an alert. By adjusting this value the user can fine-tune the detection accuracy of the algorithms. While adjusting this value for every experiment is likely to yield better results, the experiments used the same threshold for all experiments. This value was decided by following the recommendations from the respective algorithm implementation author, which was ten for USAD and four for PASAD.

5.4 Attack design

This section describes how the experiments were designed to reflect meaningful changes in a smart grid environment.

5.4.1 Network-level

At the network level, the experiments involved the manipulation of fields in the IP header. This manipulation was carried out to modify the interaction of packets among various stations within the smart grid environment. By adjusting the IP header fields, such as source and destination addresses, total length and TTL, the behaviour of the network components can be influenced. This allows for the simulation of diverse scenarios, including packet routing anomalies, DoS attacks, or unauthorised access attempts.

5.4.2 OT-level

In crafting experiments targeting the OT level, particularly focusing on SCADA and MQTT systems, other manipulation methods were used. These involved modifying the data encapsulated within the OT processes, such as sensor measurements and control commands. By altering these packaged sensor values, various behaviours within the smart grid can be simulated. This may include spoofing sensor readings, injecting false commands, or disrupting the integrity of data transmission. With these modifications, the experiments aimed to emulate potential attacks on the physical processes in smart grid infrastructure.

For each OT-level attack, specific IOAs and message id were chosen. This selection allowed for the examination of particular sensor measurements, such as current or voltage readings, and an assessment of algorithm reactions. To ensure a comprehensive dataset for both training and testing, the message id and IOA with the highest number of rows were selected for each manipulation.

5.5 Hardware

This section provides an overview of the desktop computer and Rasberry Pi 400 utilised in the experiments, along with a review of their specifications.

5.5.1 Desktop computer

All experiments conducted in this thesis were executed on a desktop computer, with detailed specifications provided in Appendix A.3. While the algorithms are ultimately intended for deployment on IoT devices, evaluating the algorithms on a more powerful device allows for more experiments to take place as the training and inference time of the algorithms is heavily decreased. Running the bulk of the experiments on a powerful computer allows for a higher degree of experimentation of various parameters without being constrained by time.

The evaluation using the more powerful desktop computer was aimed at the iterative design of experiments while finding optimal parameters and resulting evaluation metrics.

5.5.2 Raspberry Pi 400

Throughout the thesis, emphasis was placed on the development of lightweight algorithms, a concept central to the evaluation of PASAD and USAD. In this context, as discussed in Section 4.1, lightweight refers to algorithms that could be efficiently executed on resource-constrained IoT devices.

The term IoT encompasses a broad spectrum of devices, ranging from simple sensors to advanced computing platforms. Given the diversity of IoT devices, careful consideration was given to selecting an appropriate IoT device for conducting experiments and evaluating the performance of USAD and PASAD. The chosen IoT device was the Rasberry Pi 400, which is at the top end of the spectrum of IoT devices from a performance standpoint, featuring a quad-core 64-bit processor [59]. Further specifications of the device are available in Table 5.1.

The selection of the Raspberry Pi 400 as the IoT unit was motivated by the following requirements:

- **Limitations in algorithm library compatibility**: The libraries of the PASAD implementation were only compiled to the v8 version of the ARM processor, which heavily limited the selection.
- **Ease of use and implementation**: The device was chosen for its ease of use and accessibility. It requires no additional expertise to set up and use, which can be the case for other more constrained devices.

The Raspberry Pi 400 was used to determine the resource utilisation of the algorithms using the optimal parameters found using more powerful hardware.

Hardware	Rasberry Pi 400
Processor	4-core Cortex-A72 (ARM v8) 64-bit SoC @ 1.8GHz
Memory	4GB LPDDR4-3200
GPU	Broadcom VideoCore VI
Storage	MicroSD 16GB

Table 5.1: Specifications of the Raspberry Pi used in the experiments.

5.6 Selection of metrics

Metrics such as TP, TN, FP and FN, described in Section 3.4.2, do not take surrounding data into account. Since the datasets explored in this thesis include time-series data, where data points are highly dependent on each other, the impact of a specific metric varies significantly depending on its surrounding data. Therefore, time-series metrics were used instead, as they offer more meaningful insights into the outcomes generated by the algorithms. The time metrics used in the experiments of this thesis are defined below:

- **Detection delay**: The detection delay calculates the mean number of data points between the start of an attack and the time of the first detection.
- **Detection Accuracy**: The detection accuracy represents the ratio of all attacks that are detected by the algorithm.
- **False Alarms**: The false alarms represent the number of independent alert instances where the algorithm erroneously raises an alarm.
- **Attack Prediction Quality**: Represents the overall quality of the predictions in all attack windows, defined as the start to the end of an attack. More specifically, using the departure/anomaly score as the prediction value, it calculates a percentage of the mean prediction value in the attack windows in the range between the detection threshold and the mean prediction value of non-attack windows. This metric effectively allows for quantifying the quality of the predictions produced by an algorithm even though an alarm is not triggered. This metric is defined in more detail in Appendix A.1.
- **eTaPR**: The eTaP, eTaR and F1 metrics are defined in Section 3.4.2, specify time series-aware precision, recall and F1 scores to evaluate the algorithms. These metrics quantify the ability of the algorithm to accurately predict an attack within the limits of the attack window.

Furthermore, to evaluate the demand on the hardware, the algorithm implementations were slightly modified to record the utilisation of the CPU and system memory during training and inference. Notably, the PASAD and USAD implementations are written in different programming languages, C and Python respectively. These might impact the results as different levels of optimisation are possible.

While PASAD runs training and inference on a single data point at a time, USAD runs inference on a batch of windows which includes various data points and training on the entire training set at a time. Due to limitations of the USAD implementation, it was not viable to run it on a single data point at a time, without major code refactoring, which was outside the scope of this thesis. Also worth noting is that the algorithms are evaluated offline, the resource utilisation of online inference might look different.

The hardware metrics used are defined below:

- **CPU utilisation**: A percentage of the mean processor utilisation in the system. This allows to see how CPU efficient the algorithms are and whether the bottleneck is on the CPU or somewhere else. This gives some indications of how even more restrained devices might perform.

- **Memory utilisation**: The maximum amount of system memory that the algorithm utilises. During the evaluation, this metric is divided into three different components: total memory utilisation, overhead memory utilisation and the difference between the two. The overhead memory is recorded to remove as much bias as possible from the specific adaptation of the algorithm implementation. Thus, it is possible to only analyse the memory from the actual training and inference stage on the respective algorithms. As the maximum amount of memory required tends to be stable, this metric gives a good idea of the amount of memory needed to run inference on an arbitrary device.
- **Model size**: The size of the trained model inside the system. This indicates how much disk space the device needs to accommodate for each model.
- **Inference Time**: The time it takes to train the model or run inference using the trained model.

6

Experiments & Results

The designed attacks explored different aspects of OT-level and Network-level based attacks, to gain an understanding of the range of attacks they could effectively mitigate and those posing greater challenges. Each attack underwent testing with different sets of parameters to analyse their impact on the outcome.

As discussed in Section 3.2, related research identified several attacks that commonly target smart grids. The focus of the experiments was on attacks that were not only common in smart grids but also interconnected with AD systems and protocols such as MQTT, IEC 60870-5-104 and IPv4.

6.1 OT-level attacks

In the following section, attacks were conducted on the MQTT and the IEC 60870-5-104 protocol, which in this section is simply referred to as SCADA. Both protocols facilitate the transmission of sensor measurements across the grid infrastructure.

As described in Section 3.5.1, MQTT sends publish messages containing various critical information for network operations. One common message type in the dataset is readings from substations in the smart grid. This message typically includes an identifier and a signal represented by a sensor measurement or a comparable value.

In the SCADA protocol, as described in Section 3.5.2, sensor measurements are encapsulated in ASDU data units, typically containing functions like regular addresses, IOAs, and sensor values. This transmitted data contains important measurements sent to the control centre.

In the following sections, different attacks targeting MQTT and SCADA are discussed. These include frozen value attacks, incremental value attacks and a scale attack. In the frozen value attacks, sensor values were frozen across multiple attack rows, while incremental value attacks involved gradually incrementing sensor measurements over time by a specific value. The last attack performed on the OT-level data was a scale attack on the MQTT protocol, where the true sensor measurements were modified by incrementally scaling the values over time to avoid detection.

6.1.1 Frozen value attack targeting MQTT

The frozen value attack exploited vulnerabilities within the MQTT protocol, specifically manipulating MQTT publish messages. The data was sorted based on the GRP8 Current B measurement values, with message id MF46.

Manipulating the Current B parameter within the MQTT publish message can result in significant consequences. By freezing these measurement values, an attacker can transmit inaccurate data regarding the current state of the grid. Such misinformation can result in bad decision-making, leading to improper power distribution and compromised grid stability.

For this attack, two sizes of attack rows were tested. The first with 1,000 rows can be observed in Figure 6.1, where the red line between 300,000 and 350,000, indicates the manipulated changes in the data. The second attack, containing 10,000 rows, is shown in Figure 6.2, where the red line illustrates the attack.

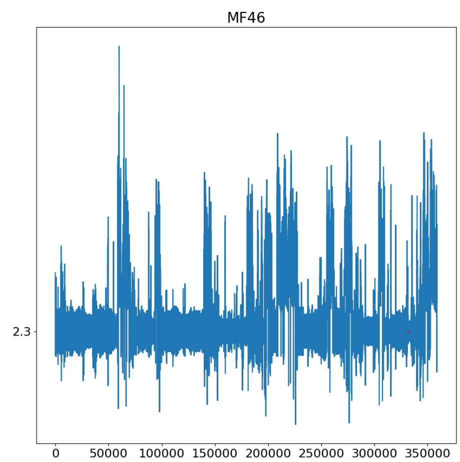

Figure 6.1: Frozen value attack on the MQTT protocol, with a red line indicating the 1,000 attack rows.

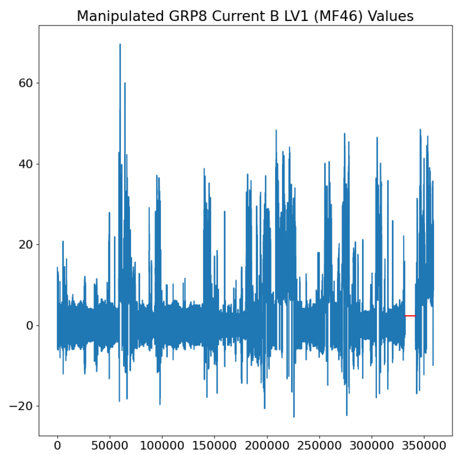

Figure 6.2: Frozen value attack on the MQTT protocol, with a red line indicating the 10,000 attack rows.

PASAD

The results from the frozen value attack, manipulating the MQTT measurement values are shown in Table 6.1. PASAD successfully detected all performed attacks with a detection accuracy and prediction quality of one, without any false alarms. For the attack with 1,000 rows, a lag of 1,000 yielded the best results, showing the lowest detection delay and the highest F1 score. The result is illustrated in Figure 6.3, demonstrating that PASAD detects the attack at the location where the true value is deployed. Similarly, for 10,000 rows, shown in Figure 6.4, a correlation was observed where larger lag sizes corresponded to poorer results in terms of both detection delay and F1 score.

Frozen value attack PASAD							
Attack rows		1,000			10,000		
Training points		200,000			200,000		
Lag		1,000	1,500	2,000	1,000	1,500	2,000
Detection Delay		72	99	107	72	99	107
Detection Accuracy		1	1	1	1	1	1
False Alarms		0	0	0	0	0	0
Prediction Quality		1	1	1	1	1	1
eTaPR	eTaP	0.7475	0.6940	0.6593	0.9557	0.9363	0.9177
	eTaR	0.9646	0.9514	0.9475	0.9964	0.9950	0.9947
	F1	**0.8423**	0.8026	0.7776	**0.9757**	0.9648	0.9546

Table 6.1: The PASAD results from the frozen value attack on the MQTT protocol.

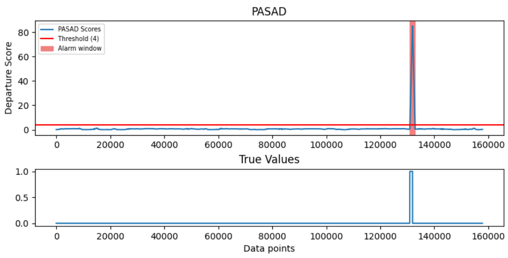

Figure 6.3: The result from the frozen value attack with 1,000 rows on PASAD.

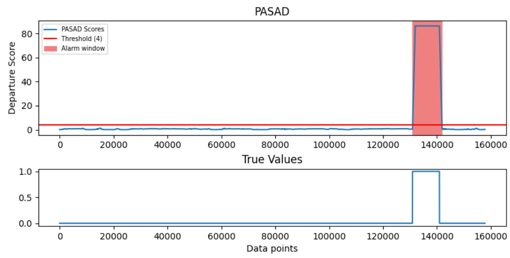

Figure 6.4: The results from the frozen value attack with 10,000 rows on PASAD.

USAD

The results from the frozen value attack, manipulating the MQTT measurement values are shown in Table 6.2. USAD managed to detect all attacks performed with a prediction quality of one, however with a large amount of false alarms, which had an impact on the F1 score. The result shows a decrease in the number of false alarms with increasing window size, although this comes at the cost of a longer detection delay. Optimal performance for USAD with 10,000 rows was achieved with a window size of 30, illustrated in Figure 6.6, yielding the best F1 score. For 1,000 rows, the best outcome in terms of false alarms, detection delay, and F1 score was observed with a window size of 30, as shown in Figure 6.5.

Frozen value attack USAD							
Attack rows		1,000			10,000		
Training points		200,000			200,000		
Window Size		12	20	30	12	20	30
Detection Delay		10	13	15	10	13	15
Detection Accuracy		1	1	1	1	1	1
False Alarms		100	53	32	88	51	24
Prediction Quality		1	1	1	1	1	1
eTaPR	eTaP	0.09523	0.1336	0.1798	0.2735	0.3662	0.4866
	eTaR	0.99028	0.9874	0.9856	0.9989	0.9987	0.9985
	F1	0.17375	0.2353	**0.3041**	0.4295	0.5359	**0.6544**

Table 6.2: The USAD results from the frozen value attack on the MQTT protocol.

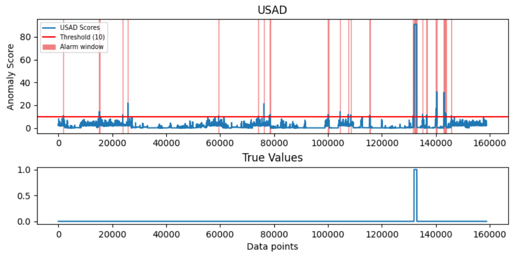

Figure 6.5: The results from the frozen value attack with 1,000 rows on USAD.

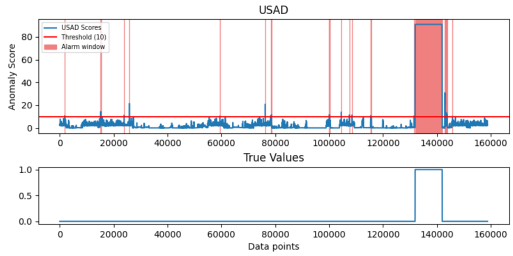

Figure 6.6: The results from the frozen value attack with 10,000 rows on USAD.

6.1.2 Incremented value attack targeting MQTT

For the incremented value attack, targeting the MQTT protocol, the data was filtered based on message id MF4, which corresponds to apparent power. Apparent power represents the total power flow within the grid and is an important parameter in smart grid management [37]. Manipulating these measurement values to appear higher than actual consumption can mislead the smart grid into increased energy usage, leading to unnecessary resource allocation.

By incrementing values gradually, attackers can evade detection for extended periods since the alterations are small enough to avoid immediate notice. Over time, these incremental changes in MQTT publish messages can cause the network to adapt to inaccurate values, potentially leading to erroneous outcomes for OT operators. As apparent power is closely tied to voltage levels in the power grid, incorrect readings can result in voltage instability, blackouts, and equipment damage.

The attack was executed using two different sets of attack rows: 500 rows, illustrated in Figure 6.7 and 5,000 rows, illustrated in Figure 6.8, each with different increment values, 0.001 and 0.01 respectively. This meant that the MQTT value was incremented by the specific increment value for each row.

PASAD

The results of the experiments on PASAD, where the slow increment attack on the MQTT protocol was performed, are shown in Table 6.3. PASAD found all the attacks, with different lag sizes and without any false alarms. In the attack involving 500 attack rows and an increment value of 0.01, it was found that using a lag size of 1,000 resulted in the highest F1 score. This is illustrated in Figure 6.9. However, this was accompanied by a longer detection delay compared to using a lag size of 1,500. A similar result was observed in the attack with 5,000 attack rows and an increment value of 0.001, where a lag of 1,000 also yielded the highest F1 score.

USAD

The results of the experiments on USAD, where the slow increment attack on the MQTT protocol was performed, are shown in Table 6.4. USAD managed to find all

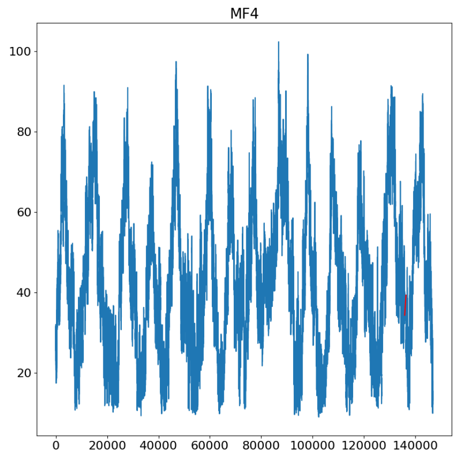

Figure 6.7: The slow increment attack on the MQTT protocol, with a red line indicating the 500 attack rows.

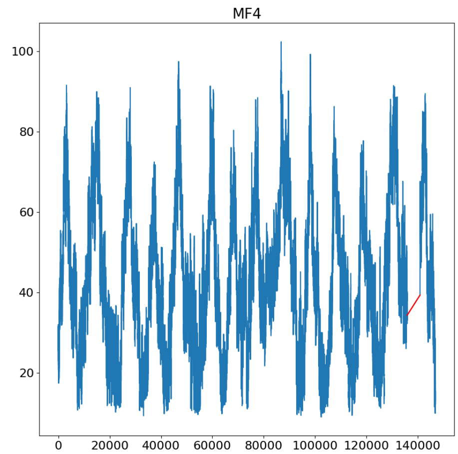

Figure 6.8: The slow increment attack on the MQTT, with a red line indicating the 5,000 attack rows.

Incremented value attack PASAD							
Attack rows		500			5,000		
Increment		0.01			0.001		
Training points		80,000			80,000		
Lag		1,000	1,500	2,000	1,000	1,500	2,000
Detection Delay		87	81	90	87	81	90
Detection Accuracy		1	1	1	1	1	1
False Alarms		0	0	0	0	0	0
Prediction Quality		1	1	1	1	1	1
eTaPR	eTaP	0.6545	0.6147	0.5893	0.9161	0.8838	0.8559
	eTaR	0.9160	0.9218	0.9131	0.9913	0.9919	0.9910
	F1	**0.7635**	0.7376	0.7163	**0.9522**	0.9347	0.9185

Table 6.3: The PASAD results from the incremented value attack on the MQTT protocol.

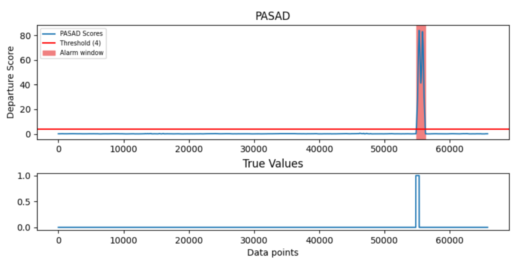

Figure 6.9: Results from the incremented value attack on MQTT with 500 rows using PASAD.

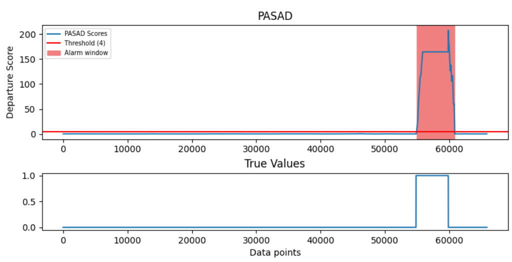

Figure 6.10: Results from the incremented value attack on MQTT with 5,000 rows using PASAD.

attacks carried out with a low detection delay.

For the attack involving 500 attack rows and an increment value of 0.01, illustrated in Figure 6.11, the optimal outcome was achieved using a window size of 30, with the best performance in terms of false alarms and F1 score. Similar results, shown in Figure 6.12, were obtained for the attack with 5,000 attack rows and an increment value of 0.001, when the window size was set to 30.

Incremented value attack USAD							
Attack rows		500			5,000		
Increment		0.01			0.001		
Training points		80,000			80,000		
Window Size		12	20	30	12	20	30
Detection Delay		12	14	16	12	14	16
Detection Accuracy		1	1	1	1	1	1
False Alarms		10	5	2	10	5	2
Prediction Quality		1	1	1	1	1	1
eTaPR	eTaP	0.42576	0.58431	0.7564	0.69829	0.80986	0.9056
	eTaR	0.98771	0.98417	0.9808	0.99881	0.99861	0.9984
	F1	0.59503	0.73327	**0.8541**	0.82194	0.89438	**0.9497**

Table 6.4: The USAD results from the incremented value attack on the MQTT protocol.

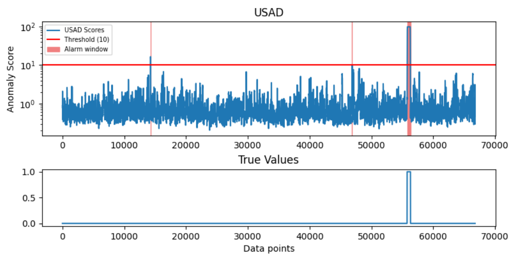

Figure 6.11: Results from slow increment attack with 500 rows using USAD.

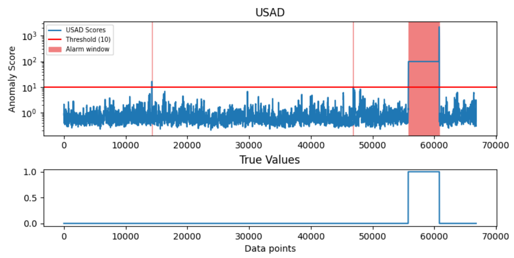

Figure 6.12: Results from slow increment attack with 5,000 rows using USAD.

6.1.3 Frozen value attack targeting SCADA

In this attack, the sensor measurements in the SCADA protocol for a specific IOA were frozen. The IOA that the data was filtered on was the GRP4 Current B measurement values. The Current B value represents the current flow in a specific part of the smart grid. Manipulating this value can misinform the SCADA system about the current flow in the system. This can result in significant safety risks such as electrical incidents and fires affecting the smart grid infrastructure. Manipulating the Current B value can also impact other parameters in the system, such as voltage, leading to instability and security risk.

The frozen value attack involved freezing 1,000 and 10,000 rows respectively, representing the number of frozen values in the dataset. Both attacks can be shown in Figure 6.13 and Figure 6.14, where the red line indicates the manipulated rows.

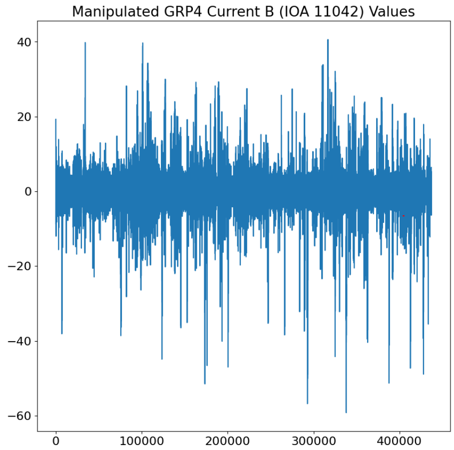

Figure 6.13: Frozen value attack targeting SCADA, with a red line indicating the 1,000 attack rows.

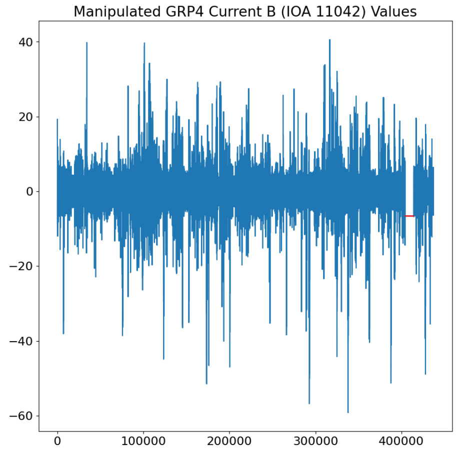

Figure 6.14: Frozen value attack targeting SCADA, with a red line indicating the 10,000 attack rows.

PASAD

The results of the experiments on PASAD, where the frozen value attack on the SCADA protocol was performed, are shown in Table 6.5. PASAD successfully identified all attacks with one false alarm each. The initial attack, comprising 1,000 attacks, yielded relatively low F1 scores. The best detection delay and F1 score were achieved with a lag of 1,500 and can be illustrated in Figure 6.15. In Figure 6.16, it is clear that PASAD performed better in the attack involving 10,000 attack rows. Specifically, a lag size of 1,000 yielded the best F1 score, although not with the lowest detection delay. As both figures illustrate, PASAD alerts prior to the attack window, resulting in its classification as a false alarm.

Frozen value attack PASAD							
Attack rows		1,000			10,000		
Training points		200,000			200,000		
Lag		1,000	1,500	2,000	1,000	1,500	2,000
Detection Delay		727	628	767	727	628	767
Detection Accuracy		1	1	1	1	1	1
False Alarms		1	1	1	1	1	1
Prediction Quality		0.6918	0.9380	0.6825	1	1	1
eTaPR	eTaP	0.3185	0.3111	0.2820	0.7218	0.6757	0.6432
	eTaR	0.6429	0.6915	0.6232	0.9637	0.9686	0.9617
	F1	0.4260	**0.4291**	0.3883	**0.8254**	0.7961	0.7708

Table 6.5: The PASAD results from the frozen value attack on the SCADA protocol.

USAD

The results of the experiments on USAD, where the frozen value attack on the SCADA protocol was performed, are shown in Table 6.6. USAD successfully detected all generated attacks but produced several false alarms. In both attacks,

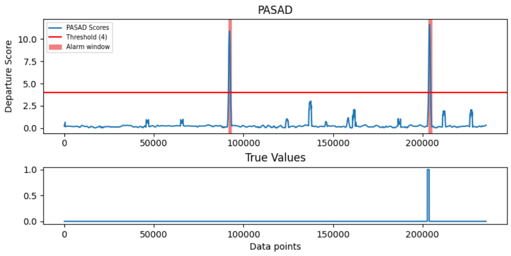

Figure 6.15: Results from the frozen value attack on SCADA with 1,000 rows using PASAD.

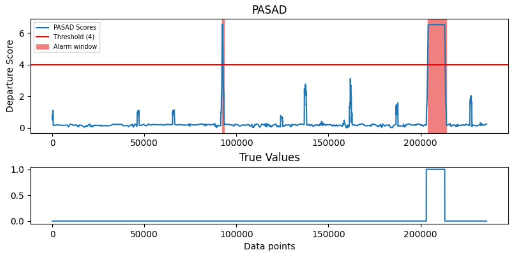

Figure 6.16: Results from frozen value attack on SCADA with 10,000 rows using PASAD.

USAD generated low F1 scores. Figure 6.17 and Figure 6.18, illustrate that USAD reacts to noise within the dataset, unrelated to the generated attack, which significantly impacts the results. Notably, for both attacks, a window size of 30 produced the best results, also resulting in the fewest false alarms.

Frozen value attack USAD							
Attack rows		1,000			10,000		
Training points		200,000			200,000		
Window Size		12	20	30	12	20	30
Detection Delay		11	13	15	11	13	15
Detection Accuracy		1	1	1	1	1	1
False Alarms		80	68	49	79	72	47
Prediction Quality		1	1	1	1	1	1
eTaPR	eTaP	0.0568	0.0594	0.0673	0.1656	0.1729	0.1933
	eTaR	0.9893	0.9869	0.9852	0.9989	0.9986	0.9984
	F1	0.1074	0.1122	**0.1260**	0.2841	0.2948	**0.3239**

Table 6.6: The USAD results from the frozen value attack on the SCADA protocol.

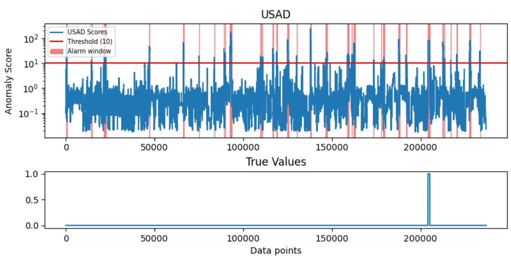

Figure 6.17: Results from the frozen value attack on SCADA with 1,000 rows using USAD.

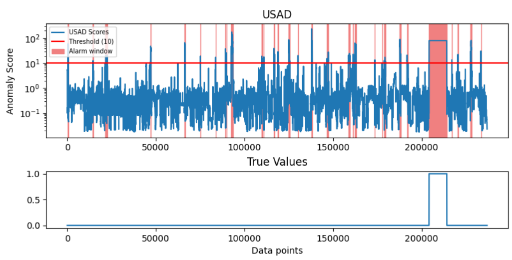

Figure 6.18: Results from the frozen value attack on SCADA with 10,000 rows using USAD.

6.1.4 Incremented value attack targeting SCADA

The incremented value attack on the SCADA protocol involved incrementing the values by a predetermined increment value. The data was filtered on IOA 11042 representing Current B. The consequences of this attack are similar to those discussed in the frozen value attack in Section 6.1.1.

In the first attack, the Current B value was multiplied by an increment factor of 0.001. Essentially, the initial value increased by 0.001 with each new row, illustrated by the red line in Figure 6.19. Then, an attack was executed with an increment value of 0.005, shown in Figure 6.20, resulting in a steeper red line. Both attacks included 5,000 attack rows and 200,000 training points.

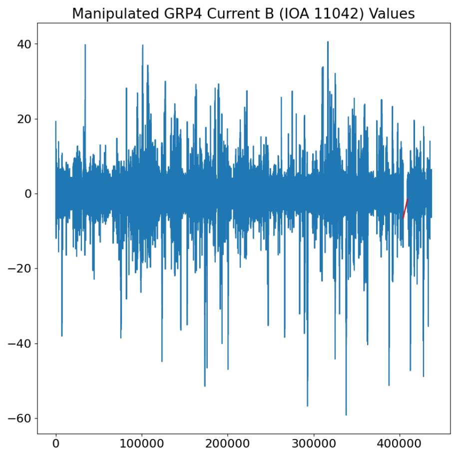

Figure 6.19: The incremented value attack on the SCADA protocol, with a red line indicating the attack with an increment value of 0.001.

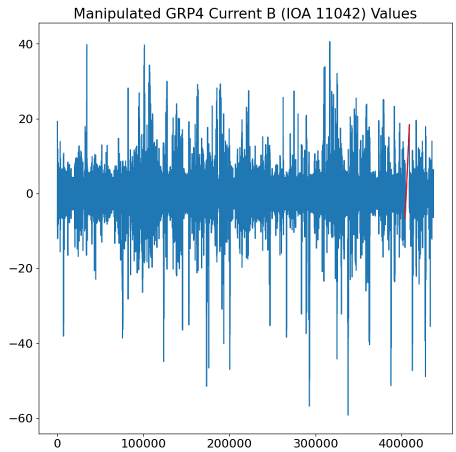

Figure 6.20: The incremented value attack on the SCADA protocol, with a red line indicating the attack with an increment value of 0.005.

PASAD

The outcome of the incremented value attack on PASAD is presented in Table 6.7. PASAD managed to detect both attacks with all but one lag configuration, yet it also generated a false alarm for each of them.

In the first attack with an incremental value of 0.001, it emerged that employing a lag size of 1,500 resulted in the best outcomes, illustrated in Figure 6.21. The figure shows the highest F1 score and shortest detection delay. Notably, PASAD failed to identify the attack with a lag size of 1,000, resulting in a detection accuracy of zero. However, a relatively high prediction quality indicated that PASAD was close to detecting the attack.

In the second attack with an increment value of 0.005, illustrated in Figure 6.22, once again, a lag of 1,500 performed the best, with the lowest detection delay and the highest F1 score.

USAD

The results from the incremented value attack on USAD are shown in Table 6.8. USAD detected all attacks with low detection delay and several false alarms. The optimal window size for both increment values was 30. Figure 6.23 and Figure 6.24 show the best result for each respective attack. As depicted in the figures, USAD's response to noise in the dataset results in several false alarms and low F1 scores.

Incremented value attack PASAD							
Attack rows		5,000			5,000		
Increment		0.001			0.005		
Training points		200,000			200,000		
Lag		1,000	1,500	2,000	1,000	1,500	2,000
Detection Delay		n/a	794	981	2937	803	994
Detection Accuracy		0	1	1	1	1	1
False Alarms		1	1	1	1	1	1
Prediction Quality		0.8275	1	1	0.9357	1	1
eTaPR	eTaP	0	0.5773	0.5337	0.5311	0.5718	0.5318
	eTaR	0	0.9208	0.9022	0.7073	0.9199	0.9009
	F1	0	**0.7097**	0.6707	0.6066	**0.7052**	0.6688

Table 6.7: The PASAD results from the incremented value attack on the SCADA protocol.

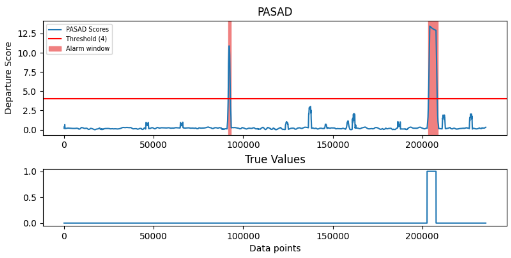

Figure 6.21: Results from the incremented value attack on SCADA with 5,000 rows and an increment value of 0.001 using PASAD.

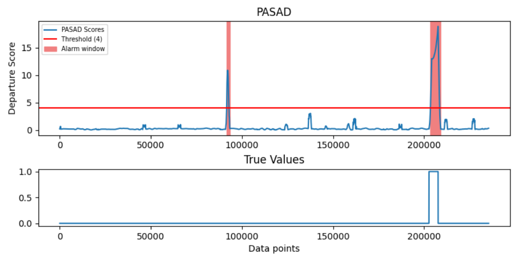

Figure 6.22: Results from the incremented value attack on SCADA with 5,000 rows and an increment value of 0.005 using PASAD.

However, a high prediction quality value indicates that USAD confidently finds the attack.

Incremented value attack USAD							
Attack rows		5,000			5,000		
Training points		200,000			200,000		
Increment		0.001			0.005		
Window Size		12	20	30	12	20	30
Detection Delay		11	13	15	10	13	15
Detection Accuracy		1	1	1	1	1	1
False Alarms		81	73	49	81	73	49
Prediction Quality		1	1	1	1	1	1
eTaPR	eTaP	0.1223	0.1260	0.1382	0.1197	0.1241	0.1382
	eTaR	0.9978	0.9974	0.9969	0.9978	0.9973	0.9969
	F1	0.2178	0.2237	**0.2428**	0.2139	0.2208	**0.2428**

Table 6.8: The USAD results from the incremented value attack on the SCADA protocol.

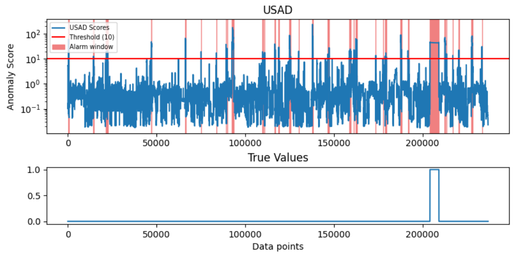

Figure 6.23: Results from the incremented value attack on SCADA with 5,000 rows and an increment value of 0.001 using USAD.

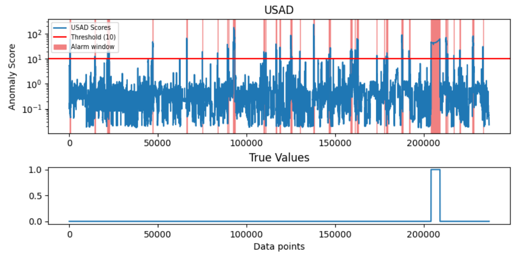

Figure 6.24: Results from the incremented value attack on SCADA with 5,000 rows and an increment value of 0.005 using USAD.

6.1.5 Scale attack targeting MQTT

The last attack performed on the OT-level data was the scale attack. Similar to the increment value attacks, the manipulated values are slowly modified over time while remaining within the normal operational range. This type of attack differs from previous ones due to its increased complexity of evading detection. It aims to disguise itself by maintaining the same behaviour in the data while slowly altering the scale of the true sensor measurements over time. This makes it very difficult for a human to detect, as the behaviour looks nearly identical to normal operation.

To avoid detection at the beginning of the attack the scale applied to the true sensor measurements starts at one. Therefore, the first values in the attack are not very harmful to the system. However, over time the scale is slowly increased and eventually reaches a level that changes the behaviour of the system. While it might seem harmless to slightly change the sensor measurements, even slight variations can put increasing strain on different physical processes in the system.

The attacks were carried out on MQTT data where the targeted message id was MF4, which correlates to apparent power, and has the same consequences as described in Section 6.1.2. In the first attack the target scale was 0.7, meaning that from the start of the attack window, the scale was one and slowly decreased over time to 0.7 and then slowly increased back to one again before the end of the attack window. The result of this manipulation is shown in Figure 6.25. The second attack had a target scale of 1.2, thus increasing the value of the sensor measurements, as shown in Figure 6.26.

PASAD

The results from the scale attack on PASAD are presented in Table 6.9. PASAD detected the attacks with all lag sizes except one, without triggering any false alarms. The best result in the attack with 20,000 rows and a scale factor of 0.7 was with a lag size of 1,500, illustrated in Figure 6.27, yielding the highest F1 score and lowest detection delay. The same lag of 1,500 produced the optimal outcome for the attack involving 20,000 attack rows and a scale factor of 1.2 which is shown in Figure 6.28. However, detection failed with a lag of 2,000, despite the prediction

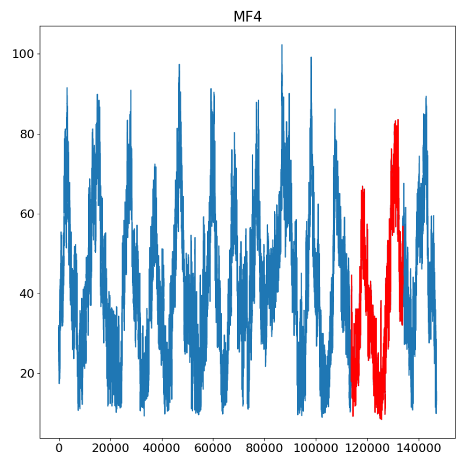

Figure 6.25: The scale attack on the MQTT protocol, with the red part indicating the manipulated data scaled with a range of [1, 0.7, 1].

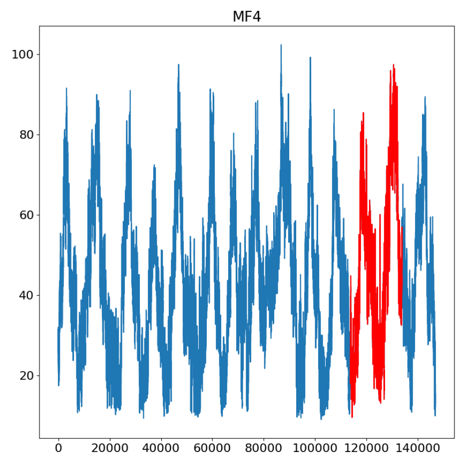

Figure 6.26: The scale attack on the MQTT protocol, with the red part indicating the manipulated data scaled with a range of [1, 1.2, 1].

quality indicating the presence of an anomaly in the data.

Scale attack PASAD							
Attack rows		20,000			20,000		
Scale Factor		0.7			1.2		
Training points		90,000			90,000		
Lag		1,000	1,500	2,000	1,000	1,500	2,000
Detection Delay		7,358	6,482	7,837	10,251	9,213	n/a
Detection Accuracy		1	1	1	1	1	0
False Alarms		0	0	0	0	0	0
Prediction Quality		0.9446	1	0.9075	0.2611	0.3304	0.2667
eTaPR	eTaP	1	1	1	1	1	0
	eTaR	0.7333	0.7656	0.7303	0.5695	0.6354	0
	F1	0.8461	**0.8673**	0.8441	0.7257	**0.7770**	0

Table 6.9: The PASAD results from the scale attack on the MQTT protocol.

USAD

The results from the scale attack using USAD are presented in Table 6.10. USAD successfully identified all conducted attacks with a detection accuracy of one. In the attack with 20,000 attack rows and a scale factor of 0.7, the best outcome, shown in Figure 6.29, was achieved with a window size of 30, resulting in the highest F1 score. However, USAD generated one false alarm. In the attack with 20,000 attack rows and a scale factor of 1.2, a window size of 20 resulted in the best outcome, achieving the highest F1 score. However, it resulted in one false alarm, as observed in Figure 6.30.

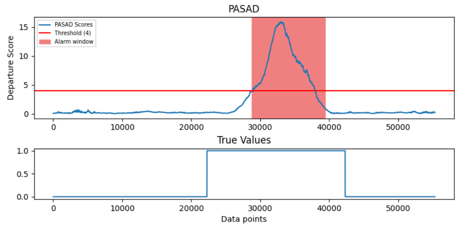

Figure 6.27: Results from the scale attack with 20,000 attack rows and a scale factor of 0.7 using PASAD with a lag of 1,500.

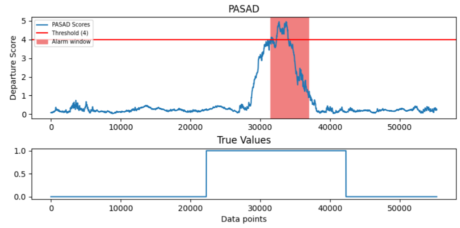

Figure 6.28: Results from the scale attack with 20,000 attack rows and a scale factor of 1.2 using PASAD with a lag of 1,500.

Scale attack USAD							
Attack rows		20,000			20,000		
Scale Factor		0.7			1.2		
Training points		90,000			90,000		
Window Size		12	20	30	12	20	30
Detection Delay		6,185	6,199	6,204	4,765	4,770	4,776
Detection Accuracy		1	1	1	1	1	1
False Alarms		6	1	1	6	1	1
Prediction Quality		0.4317	0.4002	0.4379	0.1105	0.1102	0.1100
eTaPR	eTaP	0.9629	0.9819	0.9785	0.7969	0.8949	0.8112
	eTaR	0.5527	0.5411	0.5556	0.5051	0.5695	0.5024
	F1	0.7023	0.6977	**0.7088**	0.6183	**0.7257**	0.6205

Table 6.10: The USAD results from the scale attack on the MQTT protocol.

6.2 Network-level attacks

The network-level attacks aimed to manipulate data within the IP header. The IP header encompasses various features, and manipulating each one can have different effects.

As described in Section 5.1.2, the filtered network data included two-way TCP communication between the RTU and the SCADA master or MQTT broker. The RTU is the unit that is potentially compromised. Therefore all the network-level attacks were performed on packets originating from the potentially infected RTU unit in each station.

6.2.1 TTL attack

The Time-To-Live (TTL) value is a component of the IP header and plays an important role in ensuring that the packet within the network cannot circulate indefinitely inside the network. When a packet is generated and sent across the network, it is assigned a TTL value, which operates similarly to an expiration date. Each packet is assigned a specific TTL value, and it determines how long it is permitted to exist in the network. As the packet travels through a new router, the TTL value is decremented by one, before forwarding it to the next destination. Once the TTL value reaches zero, the packet is eliminated from the network.

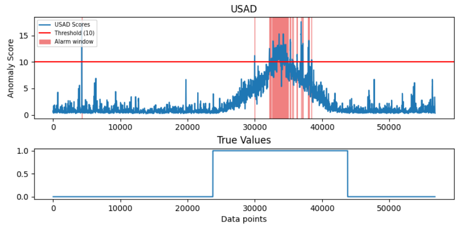

Figure 6.29: Results from the scale attack with 20,000 attack rows and a scale factor of 0.7 using USAD with a lag of 1,500.

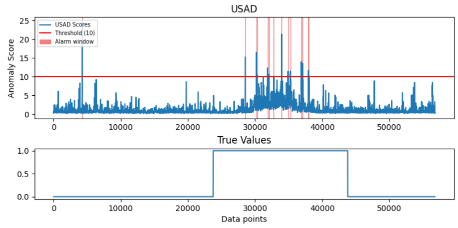

Figure 6.30: Results from the scale attack with 20,000 attack rows and a scale factor of 1.2 using USAD with a lag of 1,500.

Having both a TTL time that is too short and one that is too long can cause problems. If the TTL is too short, packets may not reach their destination before being dropped, leading to availability issues. On the other hand, having a high TTL value can overload the network, allowing packets to circulate for too long and consume resources. Manipulating the TTL value can also serve as a way for attackers to hide their tracks and evade detection by network security systems such as IDS.

When the TTL value is manipulated, it can disrupt and interfere with communication between devices and control systems in the network. This disruption can impact critical functions such as data transfer and control commands, leading to integrity issues where packets may be lost or delayed. As a result, this can result in incorrect analysis and assembly of network performance. Moreover, attackers may exploit the opportunity to intercept or disrupt communication between devices, inject malicious data, or launch other cyber attacks against the smart grid infrastructure.

The TTL experiment was executed in two ways, the static value attack and the incremental value attack. In the static value attack approach, a predefined TTL value remained constant throughout a designated number of attack rows. In the incremented value attack, the initial TTL value was incremented by one at predetermined intervals.

6.2.1.1 Static TTL value attack

The first TTL attack was the static value attack. The attack was executed by initially manipulating the TTL value at 130 for 150 rows, as illustrated in Figure 6.31, followed by manipulating it at 150, as illustrated in Figure 6.32 for 1,000 rows. Both PASAD and USAD were trained at three million training points.

PASAD

The results of the static value attack using PASAD are presented in Table 6.11. PASAD was tested with lag sizes of 500, 1,000 and 1,500. In the table, it is observed that PASAD failed to detect the attack with 500 attack rows, achieving a detection accuracy of zero. However, it is noteworthy that in Figure 6.33 PASAD detected anomalies in the data, but the departure score remained below the threshold of four, thus failing to trigger an alert. In the attack with 1,000 attack rows, PASAD

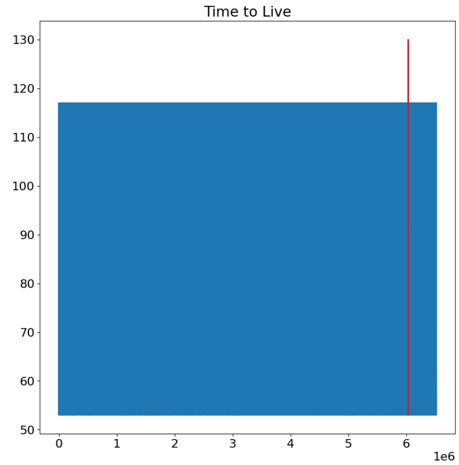

Figure 6.31: The static TTL value attack with 500 attack rows and a value of 130, contrasted against the regular TTL values inside the normal range.

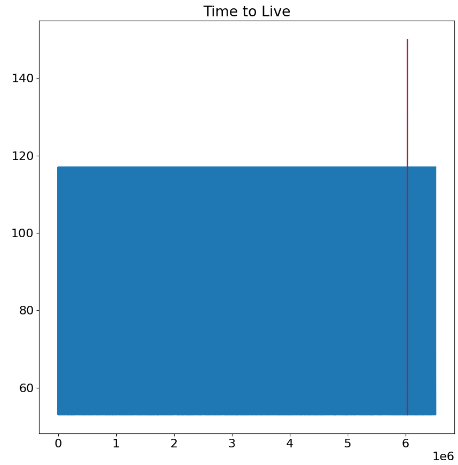

Figure 6.32: The static TTL value attack with 1,000 attack rows and a value of 150, contrasted against the regular TTL values inside the normal range.

managed to detect the attacks with a lag size of 500 and 1,000. The best result is shown in Figure 6.34, where a lag of 500 yielded the highest F1 value, the lowest detection delay and no false alarms. However, it is worth noting that even when the attack is detected, the departure score remains relatively low.

In the attacks where PASAD failed to detect anomalies, the prediction quality remained above zero. This suggests that while PASAD did not flag these instances as attacks, it did identify anomalies within the data, indicating potential areas of concern or further investigation.

TTL static value attack PASAD							
Attack rows		500			1,000		
Training points		3,000,000			3,000,000		
TTL value		130			150		
Lag		100	500	1,000	100	500	1,000
Detection Delay		n/a	n/a	n/a	n/a	348	960
Detection Accuracy		0	0	0	0	1	1
False Alarms		0	0	0	0	0	5
Prediction Quality		0.2559	0.3118	0.2584	0.2643	1	0.8130
eTaPR	eTaP	0	0	0	0	0.8171	0.0301
	eTaR	0	0	0	0	0.8260	0.5200
	F1	0	0	0	0	**0.8215**	0.0570

Table 6.11: The PASAD results from the TTL static value attack.

USAD

The outcomes of the static value attack are presented in Table 6.12. USAD was evaluated across three window sizes of 12, 20, and 30, utilising 3,000,000 training

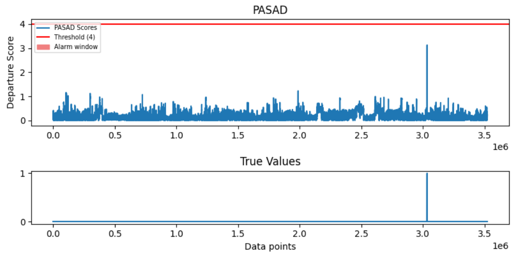

Figure 6.33: Results from the TTL static value attack with 500 attack rows and a value of 130 using PASAD with a lag of 500.

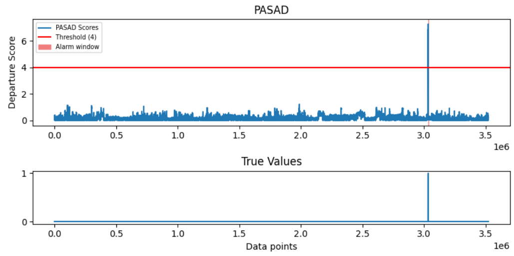

Figure 6.34: Results from the TTL static value attack with 1,000 attack rows and a value of 150 using PASAD with a lag of 500.

points. USAD failed to detect any of the conducted attacks, resulting in a detection accuracy of zero, illustrated in Figure 6.35. However, the prediction quality indicates a value above zero, suggesting that USAD detects changed behaviour in the data for all the attacks.

TTL static value attack USAD							
Attack rows		500			1,000		
Training points		3,000,000			3,000,000		
TTL value		130			150		
Window Size		8	12	20	8	12	20
Detection Delay		n/a	n/a	n/a	n/a	n/a	n/a
Detection Accuracy		0	0	0	0	0	0
False Alarms		0	0	0	0	0	0
Prediction Quality		0.1081	0.1086	0.1085	0.1081	0.1086	0.1085
eTaPR	eTaP	0	0	0	0	0	0
	eTaR	0	0	0	0	0	0
	F1	0	0	0	0	0	0

Table 6.12: The USAD results from the TTL static value attack.

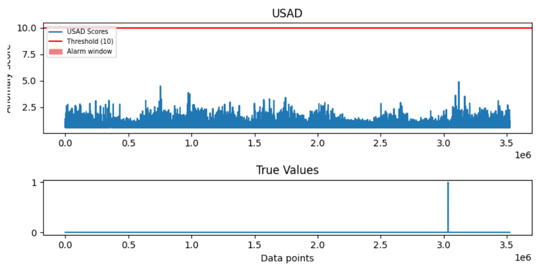

Figure 6.35: Results from the TTL static value attack with 500 attack rows and a static value of 130 using USAD with a window size of 12.

6.2.1.2 Incremented TTL value attack

The incremented TTL value attack involved incrementing the TTL value by one at specific intervals. It was tested across various sizes of attack rows, intervals and lag/window sizes to evaluate the performance of the algorithms in different scenarios.

Figure 6.36 illustrated three of the first TTL attacks with an interval of four, six and twenty, showing how the values successively increase.

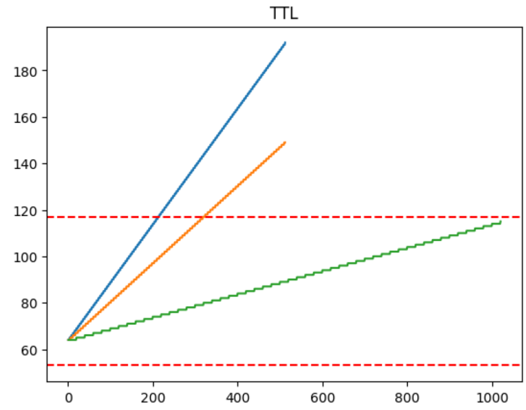

Figure 6.36: The first three out of five TTL attacks with an interval of 4, 6 and 20 respectively. The two remaining attacks are similar to the green, where the attack duration differs, however the value never increases past the maximum normal value recorded denoted by the dotted red line.

PASAD

The results of the incremental value attack on TTL values for PASAD are presented in Table 6.13. PASAD successfully detected the attacks with 1,000 attack rows with a lag of 500 or 1,000 and 10,000 attack rows with a lag size of 2,000. However, it failed to detect the remaining attacks, despite the prediction quality indicating that PASAD was close to detecting the attack. Different lag sizes were tested for each size of attack rows, but only the ones with the best results are shown in the table.

Figure 6.37 shows the results for 1,000 attack rows with a lag of 500, where PASAD successfully detects the attack. Similarly, Figure 6.38 shows the results for 10,000 attack rows with a lag of 2,000. An interesting finding was that PASAD achieved a higher F1 score and prediction quality with 10,000 attack rows compared to 1,000 attack rows. However, it generated a lower departure score and barely managed to exceed the threshold of four.

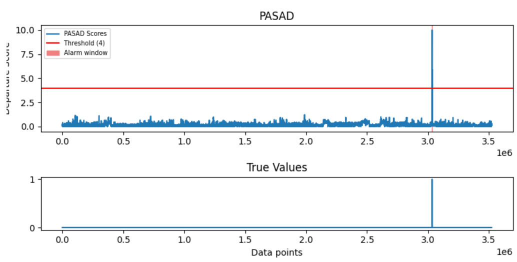

Figure 6.37: Results from the TTL incremented value attack with 1,000 attack rows and an interval of 4 using PASAD with a lag of 500.

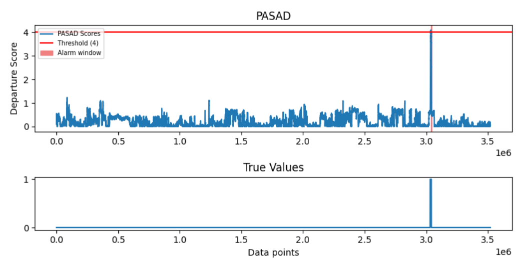

Figure 6.38: Results from the TTL incremented value attack with 10,000 attack rows and an interval of 20 using PASAD with a lag of 2,000.

Incremented TTL value attack PASAD							
Training points		3,000,000					
Attack rows		1,000	1,000	1,000	2,000	5,000	10,000
Interval		4	4	6	20	20	20
Lag		500	1,000	500	500	2,000	2,000
Detection Delay		815	967	n/a	n/a	n/a	5,211
Detection Accuracy		1	1	0	0	0	1
False Alarms		0	0	0	0	0	0
Prediction Quality		0.4651	0.2584	0.2592	0.2592	0.4924	0.7398
eTaPR	eTaP	0.6491	0.5177	0	0	0	0.8788
	eTaR	0.5925	0.5165	0	0	0	0.7394
	F1	**0.6195**	0.5171	0	0	0	**0.8031**

Table 6.13: The PASAD results from the TTL incremented value attack.

USAD

The results of the incremented TTL value attack for USAD are shown in Table 6.14. USAD failed to detect any of the attacks conducted, as shown in Figure 6.39. However, the prediction quality indicates a value above zero, suggesting that USAD detects changed behaviour in the data for all the attacks.

Incremented TTL value attack USAD							
Training points		3,000,000					
Attack rows		1,000	1,000	1,000	2,000	5,000	10,000
Interval		4	4	6	20	20	20
Window Size		12	20	12	12	12	12
Detection Delay		n/a	n/a	n/a	n/a	n/a	n/a
Detection Accuracy		0	0	0	0	0	0
False Alarms		0	0	0	0	0	0
Prediction Quality		0.1086	0.1085	0.1086	0.1098	0.1086	0.1091
eTaPR	eTaP	0	0	0	0	0	0
	eTaR	0	0	0	0	0	0
	F1	0	0	0	0	0	0

Table 6.14: The USAD results from the incremented TTL value attack.

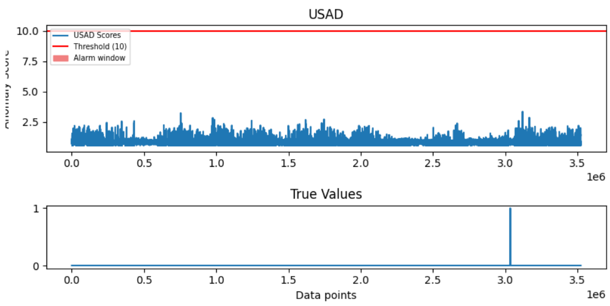

Figure 6.39: The USAD results from the incremented TTL value attack with 2000 attack rows and an interval of 20.

6.2.2 Manipulation of the total length field

This experiment targeted another field in the IP header. The primary aim was to manipulate the total length of the IPv4 packet.

Manipulating the total length field in IP headers can affect the smart grid in many ways. One consequence is the potential for the attacker to induce resource exhaustion at the receiving end of the network. Suppose the total length is set excessively large compared to the receiver capacity. In that case, it can trigger buffer overflows and memory exhaustion, making the system unresponsive or, in the worst-case scenario, causing a crash. Considering the importance of promptly detecting anomalies, attacks that target the responsiveness and efficiency of a smart grid are especially harmful.

When constructing the attack, consideration was given to the typical value of the total length. Given that the AD systems are trained on the values within the dataset, the approach involves starting from those values and introducing a larger abnormal value for the experiment.

The manipulation of the total length field consisted of two types of attacks, the static value attack and the incremented value attack.

6.2.2.1 Static total length value attack

The static total length value attack was performed using two sets of attack configurations, each with a training size of three million. This attack involved setting a specific static total length value and maintaining it constant for a defined number of attack lines.

In the first attack, illustrated in Figure 6.40, the total length was set to a static value of 800 and held constant in 10 rows. Similarly, for 100 iterations, but with a static value of 400, illustrated in Figure 6.41.

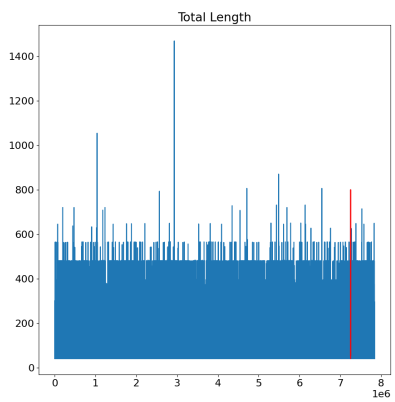

Figure 6.40: Manipulation of the total length. 10 attack rows with a static value of 800.

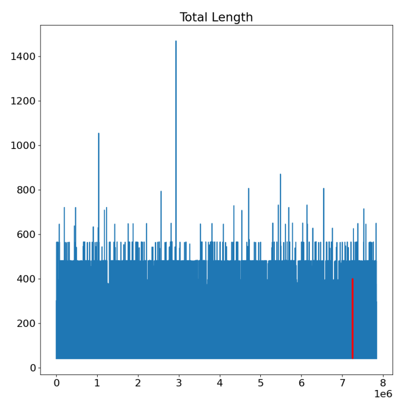

Figure 6.41: Manipulation of the total length. 100 attack rows with a static value of 400.

PASAD

The outcome of manipulating the total length of PASAD is shown in Table 6.15. The results demonstrate that PASAD successfully detected all attacks conducted with 100 attack rows, with a total length set to 400. The best result was observed with a lag size of 100, shown in Figure 6.43, resulting in the shortest detection delay and the highest F1 value. For 10 attack rows, PASAD only successfully identified the attack using the same lag configuration of 100. The result for a lag of 100 is shown in Figure 6.42, where it is worth noting that PASAD barely managed to detect the attack with a low departure score. However, it detects a change in behaviour for all three attacks, as demonstrated by the prediction quality.

In Table 6.15, the eTaPR results for 100 attack rows with a lag size of 1,000 yielded a score of zero, despite detecting the attack. This issue arises because the set lag and detection delay exceeds the number of attack rows, resulting in a score of zero in the metric.

Static total length value attack PASAD							
Attack rows		10			100		
Training points		3,000,000			3,000,000		
Total length value		800			400		
Lag		100	500	1000	100	500	1,000
Detection Delay		8	n/a	n/a	40	70	161
Detection Accuracy		1	0	0	1	1	1
False Alarms		0	0	0	0	0	0
Prediction Quality		0.4482	0.3104	0.2670	1	0.9262	0.2671
eTaPR	eTaP	0.5104	0	0	0.7142	0.5297	0
	eTaR	0.6000	0	0	0.8000	0.6500	0
	F1	**0.5515**	0	0	**0.7547**	0.5837	0

Table 6.15: Static total length value PASAD.

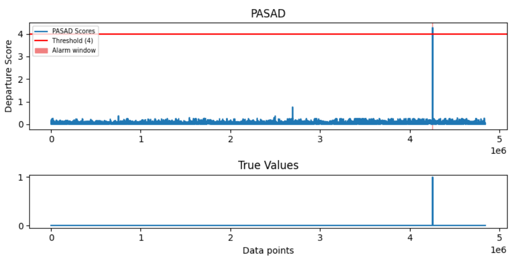

Figure 6.42: Results from the static total length attack with 10 attack rows and a value of 800 using PASAD with a lag of 100.

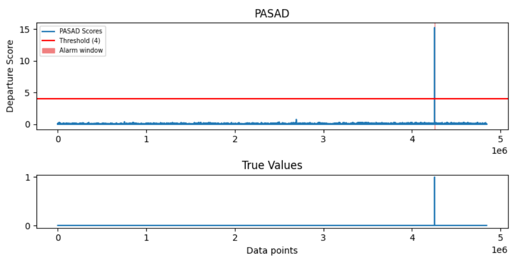

Figure 6.43: Results from the static total length attack with 100 attack rows and a value of 400 using PASAD with a lag of 100.

USAD

The result from the static total length using USAD is shown in Table 6.16. USAD managed to detect the attack with an attack size of 10 rows using a window size of 8 and 12. However, it detects a change in the behaviour of the failed attack, as demonstrated by the prediction quality. The best outcome was achieved with a window size of 8, with a low detection delay and the highest F1 score. This result is illustrated in Figure 6.44, where it was shown that USAD almost did not pass the threshold.

USAD failed to detect the attack with 100 attack rows using any window size. Figure 6.45 shows the result with a window size of 8, indicating that the algorithm responds to noise in the data and signals the presence of the attack, as indicated by the prediction quality. However, it falls short of reaching the threshold required to signal an alert.

Static total length attack USAD							
Attack rows		10			100		
Training points		3,000,000			3,000,000		
Total length value		800			400		
Window Size		8	10	12	8	10	12
Detection Delay		6	8	n/a	n/a	n/a	n/a
Detection Accuracy		1	1	0	0	0	0
False Alarms		0	0	0	0	0	0
Prediction Quality		0.6574	0.5999	0.5351	0.1938	0.1866	0.1754
eTaPR	eTaP	1	1	0	0	0	0
	eTaR	0.6176	0.5526	0	0	0	0
	F1	**0.7636**	0.7118	0	0	0	0

Table 6.16: Frozen value total length USAD.

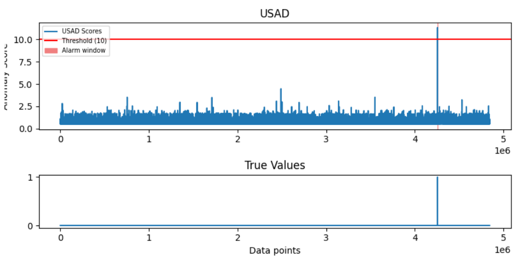

Figure 6.44: Results from the static total length attack with 10 attack rows and a value of 800 using USAD with a window size of 8.

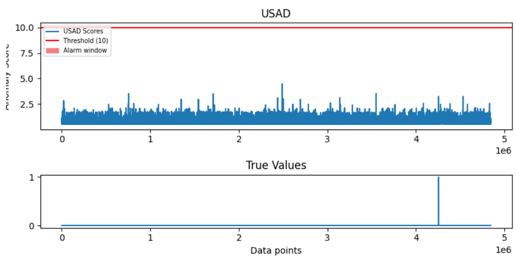

Figure 6.45: Results from the static total length attack with 100 attack rows and a value of 400 using USAD with a window size of 8.

6.2.2.2 Incremented total length value attack

The next manipulation of the total length involved gradually increasing the total length value in three distinct manners, illustrated in Figure 6.46. The total value was increased by one within a predefined interval.

The first attack comprised 5,000 attack rows, with the total length incremented by one every 15 rows. Similarly, the subsequent two attacks were designed with 10,000 rows, where the total length increased every 10 rows, and with 100,000 attack rows, where the total value increased every 40 rows.

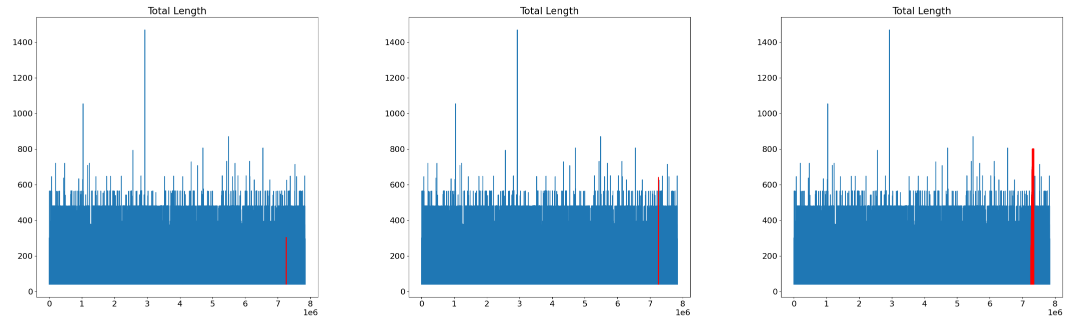

Figure 6.46: The three incremented value attacks comprised 5,000 attack rows with an interval of 15, 10,000 attack rows with an interval of 10, and the final attack involved 100,000 attack rows with an interval of 40.

PASAD

The result for PASAD after incrementing the total length values is shown in Table 6.17. For all three attacks, a lag size of 1,000 yielded the best results, displaying high F1 values. High values on the detection delay can be observed, which is a natural occurrence as the size of the attacks increases. Figures 6.47 and 6.48, respectively demonstrate good results with notably high departure scores.

Incremented total length attack PASAD							
Attack rows		5,000		10,000		100,000	
Interval		15		10		40	
Training points		3,000,000		3,000,000		3,000,000	
Lag		500	1,000	500	1,000	500	1,000
Detection Delay		4,936	2,001	2,547	1,555	11,528	6,001
Detection Accuracy		1	1	1	1	1	1
False Alarms		0	0	0	0	0	0
Prediction Quality		1	1	1	1	1	1
eTaPR	eTaP	0.5829	0.8860	0.9727	0.9482	0.9982	0.9948
	eTaR	0.5064	0.7999	0.8374	0.9222	0.9254	0.9699
	F1	0.5419	**0.8407**	0.8999	**0.9350**	0.9605	**0.9822**

Table 6.17: Incremented total length value attack using PASAD.

USAD

USAD managed to detect one of the incremented total length value attacks and the results are shown in Table 6.18. The successful detection occurred with 100,000 attack rows, with the optimal result obtained using a window size of 20, however, with high detection delays.

Figure 6.49 illustrates the results from 10,000 attack rows, indicating that USAD identifies the attack and signals the presence of an anomaly, but fails to pass the threshold for issuing an alert. Figure 6.50 demonstrates the successful detection of 100,000 attack rows, revealing a notable detection delay.

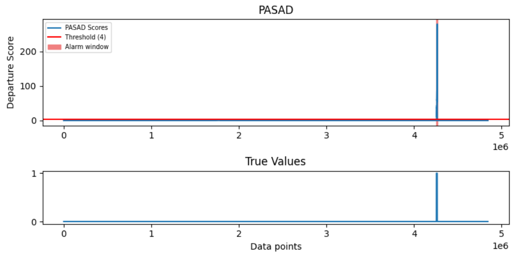

Figure 6.47: Results from the incremented total length attack with 10,000 attack rows and an interval of 10 using PASAD with a lag of 1,000.

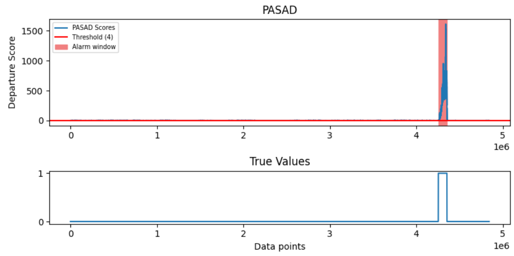

Figure 6.48: Results from the incremented total length attack with 100,000 attack rows and an interval of 40 using PASAD with a lag of 1,000.

Incremented total length attack USAD							
Attack rows		5,000		10,000		100,000	
Interval		15		10		40	
Training points		3,000,000		3,000,000		3,000,000	
Window Size		12	20	12	20	12	20
Detection Delay		n/a	n/a	n/a	n/a	41,373	43,729
Detection Accuracy		0	0	0	0	1	1
False Alarms		0	0	0	0	0	0
Prediction Quality		0.1101	0.1085	0.2436	0.2452	0.7636	0.7639
eTaPR	eTaP	0	0	0	0	1	1
	eTaR	0	0	0	0	0.7509	0.7602
	F1	0	0	0	0	0.8577	**0.8638**

Table 6.18: Incremented total length value attack using USAD.

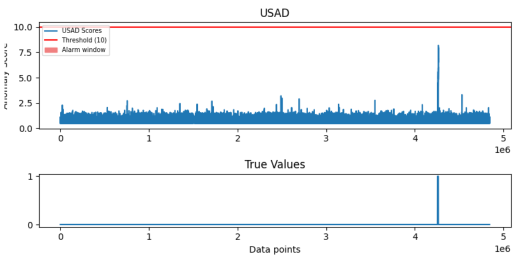

Figure 6.49: Results from the incremented total length attack with 10,000 attack rows and an interval of 10 using USAD with a window size of 20.

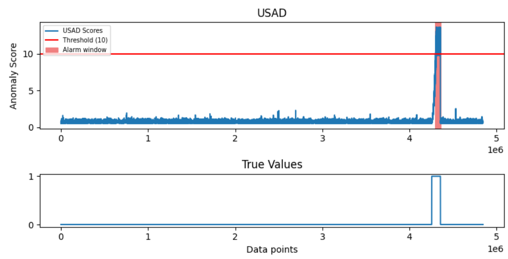

Figure 6.50: Results from the incremented total length attack with 10,000 attack rows and an interval of 40 using USAD with a window size of 20.

6.2.3 Simultaneous total length and TTL manipulation

The final type of attack performed on the network data was the combined attack. It entailed a version of a TTL attack and a packet total length manipulation attack, executed on the same dataset. Performing two attacks on the same dataset aimed to escalate the severity of the attack and explore the capability of the algorithms to detect and respond to multiple concurrent attacks.

The motivation behind an attacker decreasing the TTL value while increasing the total length can vary. Increasing the total length allows the attacker to maximise the data payload in a single packet sent over the network. This could facilitate the transmission of malicious data or code, or simply enable larger payloads to be carried in a single packet. However, manipulating the total length in this manner may trigger the security systems of a network and be detected. To evade detection, the attacker can simultaneously decrease the TTL time. This reduces the number of routers the packet passes before being dropped, giving the appearance of arriving closer to the final destination. As a result, the packet may bypass network filtering and avoid unintended routing to other destinations. By employing this tactic, the attacker can operate with greater efficiency and stealth, thereby increasing the likelihood of a successful attack while reducing the chances of detection by network defences.

6.2.3.1 Composite attack

This attack comprised two tactics employing various dataset manipulations, a TTL decrement attack and a total length increment attack.

The total length manipulation, as illustrated in Figure 6.51, involved 10,000 attack rows where the total length increased by two every 15 rows. The total length was restricted by a maximum threshold, which was set to a value of 800.

The TTL manipulation, shown in Figure 6.52, involved successive decrement of the value. This manipulation encompassed 10,000 attack rows, where the TTL value decreased by one at intervals of 150.

Both PASAD and USAD underwent testing across various lag/window sizes and with different variations of training points. The combinations yielding the best results for PASAD and USAD are present in Table 6.19.

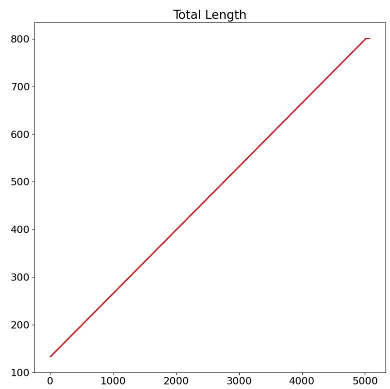

Figure 6.51: The total length manipulation with 10,000 attack rows.

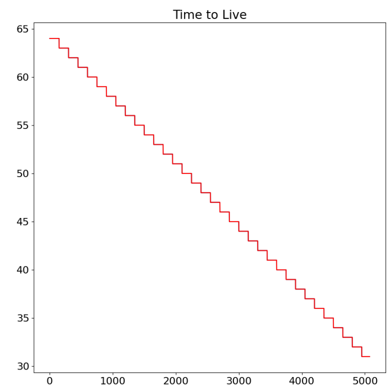

Figure 6.52: The TTL manipulation with 10,000 attack rows.

PASAD managed to find both attacks. In the attack with one million training points, a lag of 1,000 achieved the highest F1 score, shown in Figure 6.53, but with a longer detection delay compared to a lag of 500. However, PASAD demonstrated even better performance in the attack trained with three million training points as shown in 6.54, where a lag of 500 achieved the lowest detection delay and highest F1 score.

USAD successfully detected all the attacks conducted. In the attack with one million training points, the highest F1 score was achieved with a window size of 12. However, as shown in Figure 6.55, USAD gave two false alarms, along with a long detection delay. In the case of three million training points, illustrated in Figure 6.56, a window size of 8 emerged as the best option for both detection delay and F1 score.

Composite attack					
Total length Attack rows		10,000			
Increment		2			
Interval		15			
TTL Attack Rows		10,000			
Increment		-1			
Interval		150			
PASAD					
Training points		1,000,000		3,000,000	
Lag		500	1,000	500	1,000
Detection Delay		1,988	2,644	1,774	1,996
Detection Accuracy		1	1	1	1
False Alarms		0	0	0	0
Prediction Quality		1	1	1	1
eTaPR	eTaP	0.8384	0.9639	0.9764	0.9457
	eTaR	0.9006	0.8609	0.9077	0.9002
	F1	0.8683	**0.9095**	**0.9408**	0.9223
USAD					
Training points		1,000,000		3,000,000	
Window Size		8	12	8	20
Detection Delay		6,991	7,236	6,925	7,516
Detection Accuracy		1	1	1	1
False Alarms		1	2	0	0
Prediction Quality		0.5348	0.5480	0.5485	0.5335
eTaPR	eTaP	0.9949	0.9945	1	1
	eTaR	0.5995	0.6061	0.6043	0.6007
	F1	0.7482	**0.7532**	**0.7533**	0.7505

Table 6.19: Results from the single attack with one total length manipulation and one TTL manipulation using PASAD and USAD.

6.2.3.2 Repeated composite attack

The final attack conducted on the network-level data comprised four attacks within the same dataset. The experiment involved two distinct TTL manipulations and two total length manipulations.

Compared to the previous composite attack, described in Section 6.2.3.1, this attack employs a slightly different manipulation strategy. The first attack captures an initial value and increments it over time, resulting in a linear increase. In contrast,

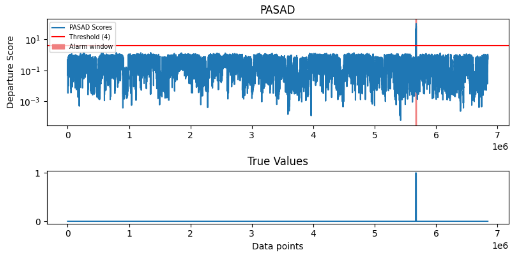

Figure 6.53: Single TTL and total length attack with 10,000 attack rows and 1,000,000 training points using PASAD with a lag of 1,000.

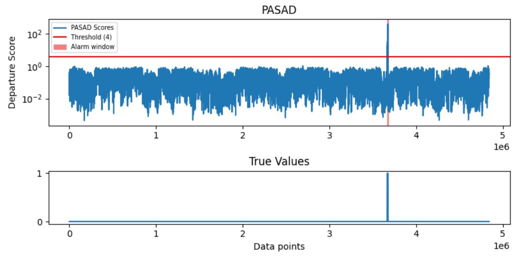

Figure 6.54: Single TTL and total length attack with 10,000 attack rows and 3,000,000 training points using PASAD with a lag of 500.

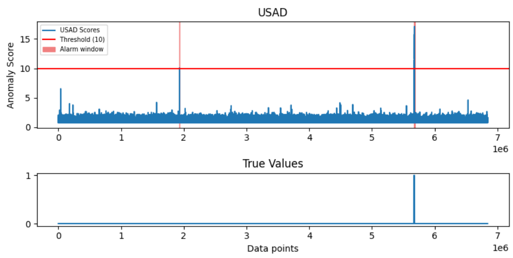

Figure 6.55: Single TTL and total length attack with 10,000 attack rows and 1,000,000 training points using USAD with a window size of 12.

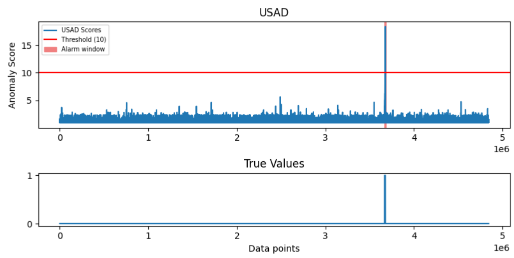

Figure 6.56: Single TTL and total length attack with 10,000 attack rows and 3,000,000 training points using USAD with a window size of 8.

this attack modifies the current value at each time point using an incremental value that changes over time, maintaining the same behaviour of the values.

The initial total length manipulation comprised 10,000 attack rows, with an incremental value that is increased by two at an interval of 35 which is added to the current value at every time point. Subsequently, the second total attack, slightly larger, consisted of 15,000 attack rows at intervals of 45. These manipulations are shown in Figure 6.57.

Both TTL manipulations involved decreasing the value by one at intervals of 170 and 250. These two manipulations are illustrated in Figure 6.58.

In both Figure 6.57 and Figure 6.58, the TTL and the total length manipulations may suggest unified attacks. However, this is not the case. The first TTL attack and the first total length attack occur simultaneously, followed by the remaining two attacks. It is important to note that each of the four attacks is distinct from the others.

PASAD successfully detected the attacks with both one and three million training points without any false alarms but with high detection delays. The results are shown in Table 6.20. In both cases, PASAD achieved optimal performance with a lag size of 500, as illustrated in Figure 6.59 and Figure 6.60.

USAD failed to detect the attacks. However, Figure 6.61 and 6.62, demonstrate

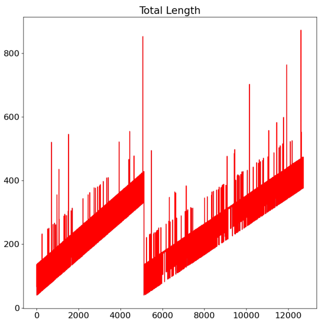

Figure 6.57: Manipulated total length values pertaining to the affected device in the attack windows.

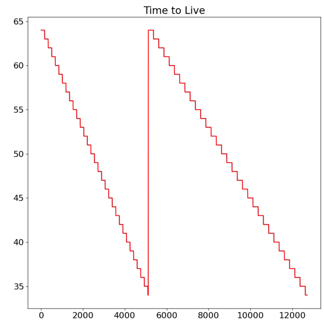

Figure 6.58: Manipulated TTL values pertaining to the affected device in the attack windows.

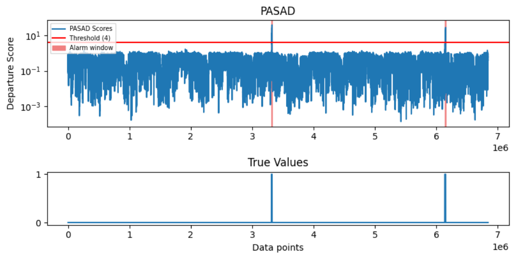

Figure 6.59: Double TTL and total length attack with 25,000 attack rows and 1,000,000 training points using PASAD with a lag of 500.

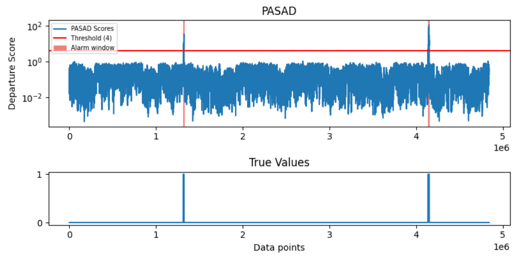

Figure 6.60: Double TTL and total length attack with 25,000 attack rows and 3,000,000 training points using PASAD with a lag of 500.

that USAD responded to both attacks, utilising window sizes of 8 and 12 with three million training points.

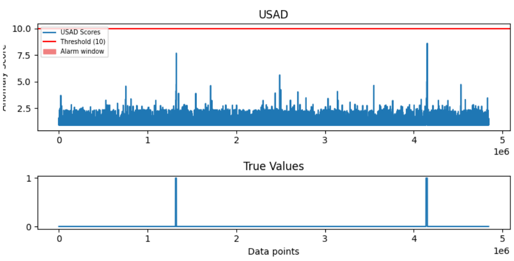

Figure 6.61: Double TTL and total length attack with 25,000 attack rows and 3,000,000 training points using USAD with a window size of 8.

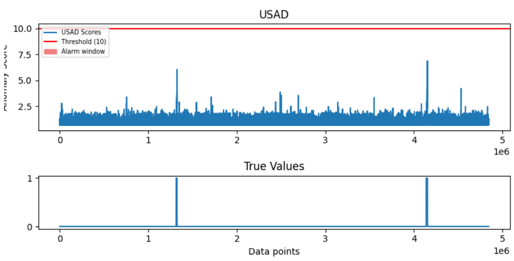

Figure 6.62: Double TTL and total length attack with 25,000 attack rows and 3,000,000 training points using USAD with a window size of 12.

Repeated composite attack					
Total length Attack 1 Rows		10,000			
Increment		2			
Interval		35			
Total length Attack 2 Rows		15,000			
Increment		2			
Interval		45			
TTL Attack 1 Rows		10,000			
Increment		-1			
Interval		170			
TTL Attack 2 Rows		15,000			
Increment		-1			
Interval		250			
PASAD					
Training points		1,000,000		3,000,000	
Lag		500	1,000	500	1,000
Detection Delay		6,101.5	7,392.0	4,075.5	4,180.5
Detection Accuracy		1	1	1	1
False Alarms		0	0	0	0
Prediction Quality		1	0.8003	1	1
eTaPR	eTaP	0.9714	0.9335	0.9793	0.9543
	eTaR	0.7208	0.6898	0.7782	0.7615
	F1	**0.8275**	0.7934	**0.8673**	0.8471
USAD					
Training points		1,000,000		3,000,000	
Window Size		8	12	8	12
Detection Delay		n/a	n/a	n/a	n/a
Detection Accuracy		0	0	0	0
False Alarms		1	0	0	0
Prediction Quality		0.1220	0.1245	0.1274	0.1170
eTaPR	eTaP	0	0	0	0
	eTaR	0	0	0	0
	F1	0	0	0	0

Table 6.20: Results from four attacks with two total length attacks and two TTL attacks using PASAD and USAD.

6.3 Resource Evaluation

The final part of the experiments relates to the resource utilisation where the algorithms were tested on the Raspberry Pi 400, as specified in Section 5.5.2. The training points and lag/window size were selected according to the best results obtained in the previous sections. The results of the evaluation for PASAD and USAD are illustrated in Figure 6.21.

For PASAD the model size remained low and stable for all experiments, ranging from 12 to 35.4 KB for the lags 500 to 1,500 for each model. PASAD proved to be capable of running both training and inference on the Raspberry Pi 400. While training, the mean CPU utilisation was high, in the range of 89.80 to 97 percent for all experiments, where a higher number of training points resulted in a higher CPU

Training Points		3,000,000			1,000,000		200,000	
Inference Points		4,843,294			5,522,394		236,924	
Features		12			15		15	
Mean Datapoint Size (Bytes)		30.55			46.98		58.52	
Algorithm		**PASAD**						
Lag		500	1,000	1,500	500	1,000	1,000	1,500
Model Size (KB)		12	23.7	35.4	12	23.7	23.7	35.4
Training	CPU Utilisation (%)	93.13	96.52	97.30	94.37	94.57	89.80	93.14
	Total Memory (MB)	7.84	17.39	32.08	7.84	17.42	17.59	32.38
	Overhead Memory (MB)	4.45	7.17	12.12	4.46	7.24	7.37	12.12
	Diff (MB)	3.39	10.22	19.96	3.38	10.18	10.22	20.26
	Time (s)	304	1130	2169	137	429	93	191
	Time/Point (μs)	101.33	376.67	723	137	429	465	955
Inference	CPU Utilisation (%)	25.05	25.03	25.02	25.05	25.01	25.68	25.47
	Total Memory (MB)	60.25	63.12	67.87	68.0	71.0	10.37	15.25
	Overhead Memory (MB)	59.87	62.75	67.62	67.62	70.62	10.12	14.87
	Diff (MB)	0.38	0.37	0.25	0.38	0.38	0.25	0.38
	Time (s)	78	137	195	92	160	7	10
	Time/Point (μs)	16.10	28.28	40.26	16.65	28.97	29.54	42.20
Algorithm		**USAD**						
Window Size		8	12	20	8	12	20	30
Model Size (KB)		399	885	2,437	399	885	2,437	5,446
Inference	CPU Utilisation (%)	76.50	78.62	85.76	75.74	81.99	81.79	82.95
	Total Memory (MB)	504.85	508.80	624.62	523.53	525.40	494.12	595.85
	Overhead Memory (MB)	485.01	464.29	494.00	482.23	487.86	344.41	362.07
	Diff (MB)	19.84	44.51	130.62	41.3	37.54	149.71	233.78
	Time (s)	490	664	898	545	740	52	90
	Time/Point (μs)	101.17	137.09	185.41	98.68	133.99	219.47	379.86

Table 6.21: Results of the resource evaluation on PASAD and USAD using the parameters that yielded the best results for the specific dataset.

utilisation. The memory utilisation without overhead fluctuated with the lag size, in the range of 3.38 to 20.26, notably it remained similar for different experiments with the same lag size. Also, the training time per data point ranged from 101.33 to 955 μs. During inference, the mean CPU utilisation was on the lower side at around 25 percent for all the different experiments. The memory utilisation without overhead showed minimal variation from 0.25 to 0.38 MB. Lastly, the inference time per data point ranged from 16.10 to 42.20 μs.

Similar, but slightly higher results were found for USAD where the model size remained stable for all experiments, ranging from 399 to 5,446 KB for the window sizes 8 to 30 for each model. Notably, the implementation of USAD did not manage to run training locally on the Raspberry Pi 400 due to memory restrictions, therefore only inference results are shown. The mean CPU utilisation was around 80 percent for all the different experiments. The memory utilisation without overhead was clearly impacted by the lag and mean data point size as it ranged from 19.84 to 233.78 MB. Finally, the inference time per data point ranged from 98.68 to 379.86 μs.

7

Discussion

This section involves discussion and reflection on the experiments performed on USAD and PASAD. One of the goals of this thesis was to investigate and evaluate the potential strengths and limitations of two state-of-the-art AD algorithms designed for IoT devices in the smart grid by using data from the same domain. Therefore, this discussion is started by looking into the results obtained after implementing and evaluating the two algorithms. This is followed by a reflection on how these algorithms performed on the selected hardware. Subsequently, a discussion addresses whether AD algorithms are a suitable choice for smart grids, considering their capabilities and limitations. Lastly, the ethical and sustainability aspects in this domain are explored.

7.1 Performance of PASAD on OT-level

PASAD demonstrated high detection performance in detecting OT-level attacks, yielding good results with high detection accuracy and prediction quality with mostly minimal detection delay and false alarms. In both the frozen value and incremented value attacks, PASAD excelled, leaving no doubt regarding its ability to detect the attacks.

However, PASAD's performance was less effective against scale attacks, where the detection delay was quite high and prediction quality scores for the second attack with 1.2 scale were low. The scale attack was the hardest OT-level attack to detect, which featured minimal behavioural deviations from the norm. The reason for this was the slowly increasing scale which was applied to the true value over time. Despite this, PASAD successfully detected the attack for most lag sizes.

One of the requirements for the security of smart grids is timely intrusion detection. PASAD exhibited relatively high detection delay values, which notably increased in line with an increased lag size. This occurs because when the lag applied to the data expands, the system requires more time to analyse the data before identifying the anomaly. This trade-off between detection sensitivity and delay is crucial to consider in such systems. While a larger lag size can enhance the capability of the system to detect anomalies by filtering out noise and minimising false positives, it may result in longer detection delays as the system waits to confirm an anomaly.

During the pre-processing phase, PASAD showed sensitivity to noise, resulting in some false alarms. Subsequent pre-processing utilising statistical features improved the accuracy of the results. However, it is worth noting that the data used in the SCADA experiments contains much noise, which poses a challenge. Therefore, PASAD may still generate false alarms in response to noisy instances, something important to handle in the upcoming use of PASAD.

Overall, PASAD demonstrates strong potential for detecting OT-level attacks, particularly in the frozen and incremented value attacks, and offers minimal training time, making it suitable for the requirements of a smart grid environment. However, its limitations must be carefully considered. Addressing these issues, enhancing noise handling and balancing detection sensitivity with response time, will be important for leveraging PASAD's full capabilities. With these adjustments, PASAD can be a strong tool for securing IoT units in smart grids against various cyber threats.

7.2 Performance of PASAD on network-level

While PASAD managed to find most network-level attacks, there was significantly more variation in the F1 scores and prediction quality for different lag sizes compared to the OT-level attacks. Also, the prediction quality was generally much lower in these experiments. Therefore finding a single lag parameter that works for most network-level attacks was difficult.

PASAD showed a great ability to detect simultaneous attacks where all attacks were detected. Interestingly, PASAD performed better in these cases with a smaller lag even though the attack rows were large. This can be due to several factors. A smaller lag size means that fewer data points are required to decide whether an anomaly is an attack, leading to fast detection. This speed is advantageous in simultaneous attacks where quick identification is important. Additionally, a smaller lag size enhances the responsiveness of the system to sudden changes making it more effective in detecting multiple concurrent anomalies rapidly.

The attack that benefitted from a larger lag size was the incremented total length attack, where a lag of 1,000 proved to be the optimal choice, which is understandable. When an attack involves making changes over time, a larger lag size allows PASAD to detect these long-term trends more effectively. This is particularly relevant to incremented value attacks, where the malicious activity gradually increases. If a lag size is too small, the algorithm might overreact to rapid, short-lived changes in the data, potentially triggered by temporary noise in the system rather than genuine malicious activity. Therefore, a larger lag size provides PASAD with a broader perspective, enabling more robust detection of actual attacks in incremented attack scenarios.

7.3 Performance of USAD on OT-level

USAD successfully detected all completed attacks on the OT level, achieving a good result with high detection accuracy and quite high prediction quality albeit with low F1 scores due to a large number of false alarms.

USAD showed low detection delay, which can be attributed to the use of small window sizes as recommended by the algorithm's creators. This configuration allowed USAD to identify anomalies, ensuring fast detection promptly. The low detection delay is a significant advantage in real-time monitoring scenarios, where responses to potential threats are important.

The occurrence of false alarms must be carefully considered. In smart grid contexts, false alarms can trigger unnecessary interventions, consume resources and undermine trust in the AD system. This can lead to delays in responding to genuine threats and disruptions in network operations and customer experience. Addressing false alarms requires additional time and resources from network operators, detracting from their focus on enhancing network performance.

One reason for these false alarms may be USAD's limitation in dealing with noise. The data utilised in the experiments were gathered at irregular intervals. This irregular sampling approach can introduce more noise into the sampled data compared to regular sampling methods, where the time intervals between measurements or observations are consistent. The resulting inconsistency in the data makes it more challenging for algorithms like USAD, which are particularly sensitive to irregularities, to identify trends and patterns accurately.

USAD shows promise as a suitable solution for smart grid OT-level security, provided that the false alarm issue is effectively managed. By improving its noise management capabilities, USAD can become a robust tool for ensuring the integrity and security of smart grid systems, balancing fast AD with good threat identification.

7.4 Performance of USAD on network-level

The results for USAD regarding network-level attacks have proven challenging to analyse. USAD exhibited inconsistent performance across the attacks, showing proficiency in some instances while failing in others.

USAD failed to detect any of the attacks involving manipulated TTL values, which are important for system security. Although USAD detected some abnormality in the data, the prediction quality was low, indicating ineffective detection. These findings suggest that USAD is not suitable for securing against these types of attacks in a smart grid environment.

However, USAD performed better in detecting the static total length attack, identifying two attacks where the manipulated value deviated most from the norm as shown in Figure 6.46 and Figure 6.40. In this case, a low window size was best for uncovering the attacks. Interestingly, USAD also detected the incremented total length value attack, but in this case, using a higher window size. The similarity between these two results is that USAD alerts when the attack indicates major deviations in behaviour, indicating its effectiveness in quickly identifying anomalies. However, the efficacy of this approach relies on the attack diverging from the typical pattern.

An interesting observation regarding USAD is its performance in simultaneous attacks, where it successfully detected the composite attack but failed to detect the repeated composite attack, despite their similar design. This outcome may be due to different factors. Firstly, the complexity of the repeated attack might have posed a significant challenge, making it difficult for USAD to distinguish between the different behaviours present in the data. Additionally, USAD may have limitations in detecting multiple attacks occurring simultaneously within the same dataset. The algorithm may not be optimised to handle such scenarios, particularly when the attacks exhibit diverse characteristics and occur concurrently. Addressing these factors, including refining algorithm parameters, enhancing training data representation, and improving the algorithm's ability to handle simultaneous attacks, could help strengthen the performance of USAD in detecting complex network-level attacks more effectively.

7.5 Constrained hardware

The resource evaluation on the Raspberry Pi 400 which is considered to be in the top range of IoT devices in terms of performance, as described in Section 5.5.2, was not aimed at comparing the two algorithms, PASAD and USAD, against each other. The reason for this is that the preconditions are not the same for both algorithm implementations. Due to the algorithms being implemented in different languages, PASAD was expected to perform better than USAD, due to the inherent performance characteristics of an application developed in C and Python.

7.5.1 Evaluation of training on the edge

Since the current state of implementation of USAD did not allow for on the edge training due to the size of the training files, only the resource evaluation of training the PASAD models were recorded. This limitation meant that this USAD implementation could not be used to leverage edge computing for training purposes.

Training the algorithms on the edge offers several advantages, primarily through more efficient handling of data. In an edge computing setup, data does not need to be transmitted to a central hub for processing. Instead, the models are trained directly on the edge devices where the data is generated. This reduces latency, decreases bandwidth usage, and enhances data privacy, as sensitive information remains localised.

PASAD works with data streams, which means that it uses one data point at a time for training or inference. Therefore, retraining a PASAD model is straightforward as the OT operator can simply restart the training process on the specific edge device. Once training is restarted, the device quickly becomes operational again, minimising downtime and ensuring continuous monitoring and anomaly detection, making PASAD cheap to deploy and require low maintenance.

PASAD proved to be very efficient at training even when the available resources are scarce where the training time per data point did not exceed 955 μs. While CPU utilisation remained high through all the experiments, this is expected as training is the most intensive process. For a device with less computational power the CPU utilisation would likely remain high but training would likely take longer. Since PASAD uses data streams for training the memory utilisation without the overhead of the implementation is minimal. The evaluations showed that this value was as low as 3.38 MB and as high as 20.26 MB. These values correspond to roughly 0.082 percent and 0.49 percent, respectively, of the total available memory of the evaluated device.

7.5.2 Inference evaluation

These algorithms proved to run more efficiently when using a separate model for different sets of features, as further discussed in Section 7.6.2. Thus, several trained models need to be stored on the IoT device which should take up minimal space. It was observed that PASAD represents an implementation with fairly optimised models in terms of storage space, where the model size for a lag of 1,000 amounted to 23.7 KB which corresponds to roughly 0.00014 percent of the total available space of the tested IoT device. Even the worst case found in the experiments where the trained model for USAD with a window size of 30 occupied just 5,446 KB which corresponds to 0.03244 percent.

For both algorithms the percentage of CPU utilisation was below 100 percent, indicating that an even more constrained device might be able to run the algorithms with similar performance.

Worth noting is PASAD's lightweight abilities related to memory. It demonstrates that without overhead the required memory is around 0.38 MB. This corresponds to only 0.009277 percent out of the total 4 GB memory. Similarly to the system memory by running several models the required memory increases linearly with the number of models deployed.

Although there are no specific ranges for the various metrics discussed to determine when an algorithm is considered lightweight, the results clearly show that these kinds of algorithms can effectively run inference per data point in under 400 μs on an IoT device with constrained hardware resources while consuming a fraction of the total available resources.

7.6 Viability of anomaly detection in smart grids

In this thesis, a key question has been whether lightweight data-driven AD algorithms are a suitable option for securing the smart grid and if they offer advantages over other available IDSs. When compared to classic ML models such as regression, the advantages of AD algorithms become apparent. Although regression models may be suitable for predictive tasks, they are not optimised for immediate AD in real-time environments.

The experiments conducted involved irregular attacks with changes over time. Such dynamic behaviour can pose challenges for signature-based IDS where algorithms may struggle to comprehend the complex and multivariate nature of the OT data. In smart grid environments, attacks may not follow pre-defined rules, therefore, AD systems are advantageous as they can detect novel attack patterns. For instance, in the scale attack, establishing pre-defined rules becomes impractical due to the complex and nuanced nature of the attack. Therefore, lightweight data-driven AD algorithms prove to be effective, especially in OT data environments, as they can detect anomalies that evolve over time.

7.6.1 Capabilities

A range of strengths and limitations in the tested algorithms have been identified through the execution of the experiments. These findings provide valuable insights into the specific properties of AD algorithms, helpful in the assessment of their suitability for integration within smart grid systems.

The evaluated algorithms have demonstrated their lightweight nature, requiring minimal resources while delivering efficient performance. This makes them a good solution for training and inference on edge devices. By leveraging edge computing, advantages such as reduced latency, bandwidth optimisation, cost efficiency, operational resilience, low maintenance, and increased privacy can be leveraged in the grid [33]. These advantages are particularly pronounced compared to rule based IDS where the OT-operators need to maintain the rule set for every type of data.

Alongside this efficiency, they have exhibited high accuracy in AD, ensuring the reliable identification of potential threats. These systems have also displayed a notable capability to detect changes in behaviour within normal ranges, offering an understanding of system dynamics. This capability is important for identifying subtle deviations that might otherwise go unnoticed.

A noteworthy finding was the ability of the system to detect stealth attacks, such as the scale attack. The systems quickly identified these attacks, often characterised by their extended duration and subtle tactics. This fast detection is important for mitigating the risks posed by an attack that exploits vulnerabilities over an extended period, as exemplified by the attack in Ukraine discussed in Section 1.2.

7.6.2 Limitations

One critical limitation identified in the AD systems is their sensitivity to noise. Throughout this thesis, this sensitivity has been a consistent issue, often leading to numerous false alarms. This has undermined the reliability of the system, making it difficult to separate between actual threats and false positives. Notably, in the experiments using SCADA data, the noise introduced by the irregular sampling of the data made accurate detection much more challenging. Thus, enhancing the ability of the systems to filter out noise will be important for improving its accuracy and reliability in future implementations.

Another limitation encountered in the AD systems was determining the optimal balance between lag/window size for effective attack detection. It became evident that varying sizes significantly influenced detection outcomes, with effectiveness varying based on the attack type and size. This was particularly noticeable in the network-level attacks where slightly different lag/window sizes had a big effect on the performance of the algorithm. Given the impact on detection accuracy, there is a need to establish guidelines for setting these parameters to achieve optimal detection results.

An important parameter to consider is the threshold [7]. In the experiments, a consistent threshold was used. This threshold significantly impacts the results, as it determines the sensitivity of the AD system. Therefore, it is important to keep in mind that different threshold values could lead to different outcomes. It is difficult to set an optimal threshold for all experiments, making it a parameter worthy of further investigation.

In the OT experiments conducted, the AD systems showed great performance. A significant observation in these attacks, as opposed to network attacks, was the use of filtered data focused on only one specific IOA and message id at a time. This approach yielded optimal performance for the systems compared to including multiple components in the dataset. It is important to note this limitation, as the performance of the system tends to decrease with an increased number of features included in the dataset. Therefore these algorithms benefit from being deployed using several models simultaneously where each model monitors a limited selection of features.

7.6.3 Recommendations

Using the insights gained from the capabilities and limitations of lightweight data-driven AD algorithms in the smart grid domain several conclusions can be drawn.

Since the performance of the system often decreases with an increased number of features in the dataset, it is recommended to use several models for each dataset, with each model focusing on specific features. This approach could help to effectively utilise the capability of filtering data based on specific IOAs and message ids, making it easier to find the correct parameters for individual threats. By adapting each model to a specific task, performance can be improved, as optimal lag sizes and thresholds can vary between different scenarios.

The AD algorithms have shown greater performance and efficiency in detecting stealth attacks than small fast attacks. Therefore, it is recommended to utilise these systems for identifying such types of attacks. If the AD algorithms are used in systems where small and quick attacks occur, they should be combined with other security measures, such as a firewall, to ensure complete protection.

Lightweight data-driven AD algorithms are easy to use and have shown high detection accuracy. For future use, it is recommended to manage noise, which will help reduce false alarms and save unnecessary time and energy for OT operators.

Finally, this type of algorithms showed the ability of efficient detection using limited resources for training and inference which allows more flexibility as the algorithms can be fully deployed on the edge without the need for a central IDS. Given the numerous advantages of leveraging edge computing, it is recommended to further explore this aspect.

7.7 Ethics & Sustainability

Given that the smart grid is a critical infrastructure it is very important to limit the disclosure of information to the public. This data can not only reveal sensitive personal information but may also give insights into the operational details of the grid which can present opportunities for exploitation.

This thesis was responsible for both designing and evaluating the experiments. Despite efforts to minimise unintended bias in the experimental design, such bias can not be eliminated and should be acknowledged. The experiments were intentionally designed to push the algorithms to their limits, often presenting challenges that were not expected to be overcome. This approach was taken to thoroughly test the capabilities and limitations of the algorithms under extreme conditions. Further work is recommended to test the algorithms using known attacks, preferably in an online setting.

When dealing with ML, it is imperative to consider the data that is fed into the system and ensure that it is not biased, since the same bias will be reflected in the results of the experiments. Data extracted from real-world power grids was chosen, but it might have included unintended biases. An example of this is the irregular sampling of CISSAN which may have different impacts on the different algorithms. Therefore, it was considered prudent to evaluate using more than one dataset, CISSAN and SWaT for fairness and completeness.

This thesis aimed to advance smart grid research by evaluating state-of-the-art lightweight algorithms and finding suitable algorithms to enable this progression. Therefore, it is essential to acknowledge that lightweight algorithms may possess inherent limitations compared to non-lightweight counterparts, which had a higher capacity to analyse data and thus achieved higher attack detection accuracy. While the algorithms presented in this thesis are viable options, it is worth considering these limitations and potentially use these algorithms as an addition to already existing security.

Finally, explainability is an ethical aspect within the smart grid domain that is becoming increasingly important as ML models and lightweight data-driven AD systems become more advanced. As the complexity increases the level of transparency in decision-making processes decreases. This warrants some consideration about accountability when the algorithms make the wrong decisions. Clearly defining accountability becomes a crucial aspect in ensuring the ethical deployment of these systems. This involves clearly understanding the potential limitations of the algorithms.

8

Event Management Guidelines

As discussed in Section 2.4, the need to manage events from AD systems is increasing. During the preparations and execution of experiments, several challenges were identified that can make it difficult for OT operators to understand the alerts generated. This section addresses the challenges encountered during the experiments and provides suggestions for addressing these challenges in future event management.

8.1 Current challenges

In this thesis, various experiments were conducted, yielding different types of outcomes. These experiments, referred to as attacks, generate alerts based on what the algorithm identifies as anomalous behaviour in the data. Instead of generating alerts for every new attack window, these algorithms generate an alert for every data point that is considered anomalous. OT operators must understand and analyse these alerts to determine the necessary measures. Several challenges have been identified during the experiments that can complicate this process. Addressing these challenges can improve the ability of OT operators to manage alerts more effectively in the future.

The initial challenge identified was the management of false alarms generated by the AD systems. False alarms present challenges for OT operators, who must discern between actual threats and false positives to prevent unnecessary disruptions to system operations.

Another challenge that emerged over time was managing system alerts that did not arrive continuously. Instead, these alerts came sporadically, especially when an attack generates multiple alerts with gaps in between, while actually belonging to the same attack window. This issue is illustrated in Figure 6.30, where the attack is detected but is split into separate alerts. This behaviour can be problematic for both the system and the OT operator. Gaps in alerts can lead to false positives or false negatives, reducing the reliability of the alert system. Additionally, obtaining a complete picture of the attack becomes difficult, which may cause operators to miss the context and analyse isolated alerts instead of continuous streams.

The algorithms investigated in this thesis only return a binary output representing an alert as well as an anomaly score. Thus, if the system were to only escalate the alerts to the human OT operator, it would be hard to distinguish what part of the system is affected and get a holistic view of the status of the system. One particular challenge arises when the input data includes a variety of features, such as the network data used in this thesis. Therefore, identifying the part of the data that the algorithms found anomalous is especially challenging.

8.2 Solution guidelines

The proposed theoretical solution for the building blocks of a simple event management system builds on the current challenges as well as specific characteristics that were recorded during the execution of the experiments in this thesis.

To manage how an operator should understand the origin and impact of system alerts, it is important to differentiate the source of the alerts and their potential consequences. This issue can be addressed by splitting functions into multiple models, each dedicated to a specific data stream. These streams can be sorted based on IP addresses and an id that describes where the model is located and what type of data it monitors. When an alert is received by the operator for a specific model, it might then become clearer what the operator is analysing.

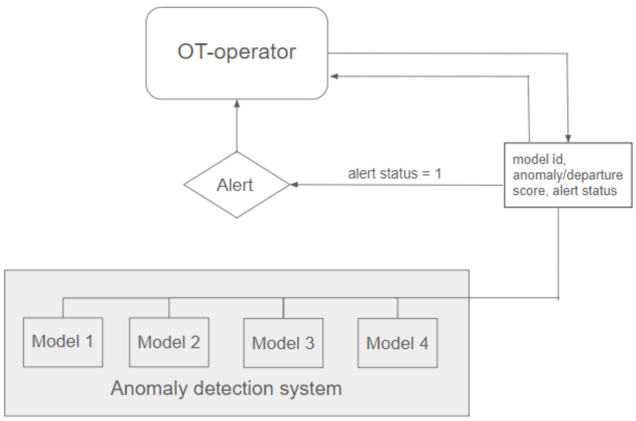

Figure 8.1: A diagram of the proposed solution for event management.

The suggested solution is illustrated in Figure 8.1 and includes the following components.

- **Log the anomaly/departure scores sent from the IoT-device**: Each time the models on the IoT-device record a new score, three values are propagated to the OT operator; the model id, anomaly/departure score and alert status. These values are then stored in a list of fixed size upon arrival for later retrieval, where the size of the list may vary based on the lag/window size of the specific model.

- **Alert sent from IoT-device**: When an alert is received, calculate a probability score that takes historical anomaly/departure scores into consideration and evaluates the probability of an attack. By using historical values, the trend can be analysed and different scenarios can be differentiated. For example, sudden temporary increases in departure scores (may be indicative of false alarms) or slowly increasing departure scores (may suggest that an attack is ongoing). The historical departure score values are aggregated and a prediction score is calculated for each model. Furthermore, the behaviour of other models can simultaneously be analysed to detect any attack correlations.

- **Display the information to the human OT operator**: As mentioned in Section 3.1, a user-friendly interface is very important and should be considered. The most important factors in this solution are the historical values, the calculated prediction score and possible related models.

These suggestions are based on observations obtained throughout this thesis and are intended only as guidelines. While these recommendations provide a starting point, a more comprehensive study is essential to fully understand and address the complexities involved.

9

Conclusion

This chapter provides suggestions for future work and summarises the key findings of this thesis.

9.1 Future work

In future work, finding a way to standardise the choice of parameters used in the algorithms, especially lag/window size and training points would be beneficial. These parameters greatly affect the results and can determine whether the attack is detected. Further investigation into how the size of the window/lag impacts the detection accuracy would provide valuable insights for improving both the usability and robustness of the system.

Various attacks were carried out in the experiments, but one of the most relevant attacks, according to related research, against the smart grid is the DoS attack. Initially, the plan was to perform such an attack, however, as discussed earlier, the data used in the experiments was collected at irregular intervals, resulting in irregular timestamps, which made it more difficult to execute a correct DoS attack. However, since this attack is common and poses significant threats, it is crucial to perform such tests on the systems to investigate how they can handle them.

Expanding the scope of testing on less advanced hardware would be beneficial, as the tests in this thesis were performed on more advanced IoT units. This would provide a more comprehensive understanding of the capacity and scalability of the systems. This could reveal potential limitations or challenges that may not be evident when using high-end devices.

Improving event-handling capabilities within AD systems is essential for their effective operation. It is important to develop strategies that allow OT operators to in an easy way read, analyse and respond to system events. There is a noticeable lack of research in this area, highlighting the need for efforts in future studies to thoroughly address this gap.

9.2 Conclusion

In this thesis, two state-of-the-art lightweight data-driven AD algorithms were selected and evaluated using a series of experiments, aimed at investigating the viability of these kinds of algorithms in smart grids with IoT components. The experiments included two different attack areas, namely OT- and network-level attacks, which were crafted by manipulating real smart grid operational data.

The selected algorithms, PASAD and USAD, were successfully able to detect various long-duration attacks with stealth characteristics, while other, shorter and more direct attacks, were significantly harder to detect. Despite this, these lightweight data-driven AD algorithms proved to be a good fit for the experiments evaluated in this thesis.

Finally, the knowledge acquired from executing the experiments was used to propose guidelines for developing an event management system that handles alerts produced by different models to provide valuable and actionable information to the OT operator.

Bibliography

[1] Uttam Adhikari, Shengyi Pan, and Tommy Norris. Industrial control system (ics) cyber attack datasets. `https://sites.google.com/a/uah.edu/tommy-morris-uah/ics-data-sets`, 2014. Accessed on March 4, 2024.

[2] Mukhtar Ahmad. *Power System State Estimation.* Artech House, 2012. ISBN 9781608075126. URL `https://search.ebscohost.com/login.aspx?direct=true&db=cat07472a&AN=clec.EBC1115667&site=eds-live&scope=site&authtype=guest&custid=s3911979&groupid=main&profile=eds`.

[3] Chuadhry Mujeeb Ahmed and Nandha Kumar Kandasamy. A comprehensive dataset from a smart grid testbed for machine learning based cps security research. In Habtamu Abie, Silvio Ranise, Luca Verderame, Enrico Cambiaso, Rita Ugarelli, Gabriele Giunta, Isabel Praça, and Federica Battisti, editors, *Cyber-Physical Security for Critical Infrastructures Protection*, pages 123–135, Cham, 2021. Springer International Publishing. ISBN 978-3-030-69781-5.

[4] Massimiliano Albanese, Robert F. Erbacher, Sushil Jajodia, C. Molinaro, Fabio Persia, Antonio Picariello, Giancarlo Sperlì, and V. S. Subrahmanian. *Recognizing Unexplained Behavior in Network Traffic*, pages 39–62. Springer New York, New York, NY, 2014. ISBN 978-1-4614-7597-2. doi: 10.1007/978-1-4614-7597-2_3. URL `https://doi.org/10.1007/978-1-4614-7597-2_3`.

[5] Fabrizio Angiulli, Fabio Fassetti, and Luca Ferragina. Reconstruction error-based anomaly detection with few outlying examples, 2023.

[6] Wissam Aoudi and Magnus Almgren. A scalable specification-agnostic multi-sensor anomaly detection system for IIoT environments. *International journal of critical infrastructure protection*, 30:100377, 2020.

[7] Wissam Aoudi and Magnus Almgren. A framework for determining robust context-aware attack-detection thresholds for cyber-physical systems. In *Proceedings of the 2021 Australasian Computer Science Week Multiconference*, ACSW '21, New York, NY, USA, 2021. Association for Computing Machinery. ISBN 9781450389563. doi: 10.1145/3437378.3437393. URL `https://doi.org/10.1145/3437378.3437393`.

[8] Wissam Aoudi, Mikel Iturbe, and Magnus Almgren. Truth will out: Departure-based process-level detection of stealthy attacks on control systems. In *Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security*, CCS '18, page 817–831, New York, NY, USA, 2018. Association for Computing Machinery. ISBN 9781450356930. doi: 10.1145/3243734.3243781. URL https://doi.org/10.1145/3243734.3243781.

[9] Wissam Aoudi, Albin Hellqvist, Albert Overland, and Magnus Almgren. A probe into process-level attack detection in industrial environments from a side-channel perspective. In *Proceedings of the Fifth Annual Industrial Control System Security (ICSS) Workshop*, ICSS, page 1–10, New York, NY, USA, 2019. Association for Computing Machinery. ISBN 9781450377195. doi: 10.1145/3372318.3372320. URL https://doi.org/10.1145/3372318.3372320.

[10] Apriorit. Internet of Things (IoT) Security: Challenges and Best Practices. https://www.apriorit.com/white-papers/513-iot-security, 2022. Accessed: 2023-11-30.

[11] Julien Audibert, Pietro Michiardi, Frédéric Guyard, Sébastien Marti, and Maria A. Zuluaga. Usad: Unsupervised anomaly detection on multivariate time series. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, KDD '20, page 3395–3404, New York, NY, USA, 2020. Association for Computing Machinery. ISBN 9781450379984. doi: 10.1145/3394486.3403392. URL https://doi.org/10.1145/3394486.3403392.

[12] Shampa Banik, Sohag Kumar Saha, Trapa Banik, and S M Mostaq Hossain. Anomaly detection techniques in smart grid systems: A review. *2023 IEEE World AI IoT Congress (AIIoT), AI IoT Congress (AIIoT), 2023 IEEE World*, pages 0331 – 0337, 2023. ISSN 979-8-3503-3761-7. URL https://search.ebscohost.com/login.aspx?direct=true&db=edseee&AN=edseee.10174485&site=eds-live&scope=site&authtype=guest&custid=s3911979&groupid=main&profile=eds.

[13] Mansi Bhavsar, Kaushik Roy, John Kelly, and Odeyomi Olusola. Anomaly-based intrusion detection system for IoT application. *Discover Internet of Things*, 3(1):5, May 2023. ISSN 2730-7239. doi: 10.1007/s43926-023-00034-5. URL https://doi.org/10.1007/s43926-023-00034-5.

[14] Gordon Clarke and Deon Reynders. *Practical Modern SCADA Protocols : DNP3, 60870. 5 and Related Systems.* Elsevier Science & Technology, 2004. ISBN 9780080480244. URL https://search.ebscohost.com/login.aspx?direct=true&db=cat07472a&AN=clec.EBC226682&site=eds-live&scope=site&authtype=guest&custid=s3911979&groupid=main&profile=eds.

[15] Clavister. Clavister wins european research project to make power grids safer. https://www.clavister.com/

clavister-wins-european-research-project-to-make-power-grids-safer/, 2022. Accessed on April 8, 2024.

[16] Cloudflare. What is Mirai? https://www.cloudflare.com/learning/ddos/glossary/mirai-botnet/, 2023. Accessed: 2023-11-29.

[17] Cloudflare. What is a denial-of-service (dos) attack? https://www.cloudflare.com/learning/ddos/glossary/denial-of-service/, N.D. Accessed on February 22, 2024.

[18] Andrew A. Cook, Göksel Mısırlı, and Zhong Fan. Anomaly detection for iot time-series data: A survey. *IEEE Internet of Things Journal*, 7(7):6481–6494, 2020. doi: 10.1109/JIOT.2019.2958185.

[19] Mohamed Endi, YZ Elhalwagy, et al. Three-layer plc/scada system architecture in process automation and data monitoring. In *2010 the 2nd international conference on computer and automation engineering (ICCAE)*, volume 2, pages 774–779. IEEE, 2010.

[20] European Commission. Smart grids and meters. https://energy.ec.europa.eu/topics/markets-and-consumers/smart-grids-and-meters_en, N.D. Accessed on April 15, 2024.

[21] Cheng Feng and Pengwei Tian. Time series anomaly detection for cyber-physical systems via neural system identification and bayesian filtering. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*, KDD '21, page 2858–2867, New York, NY, USA, 2021. Association for Computing Machinery. ISBN 9781450383325. doi: 10.1145/3447548.3467137. URL https://doi.org/10.1145/3447548.3467137.

[22] Clement Fung, Eric Zeng, and Lujo Bauer. Attributions for ML-based ICS anomaly detection: From theory to practice. In *Proceedings of the 31st Network and Distributed System Security Symposium*. Internet Society, February 2024. To appear.

[23] Daniel Fährmann, Naser Damer, Florian Kirchbuchner, and Arjan Kuijper. Lightweight long short-term memory variational auto-encoder for multivariate time series anomaly detection in industrial control systems. *Sensors*, 22(8), 2022. ISSN 1424-8220. doi: 10.3390/s22082886. URL https://www.mdpi.com/1424-8220/22/8/2886.

[24] Geeksforgeeks. Confusion matrix in machine learningr. https://www.geeksforgeeks.org/confusion-matrix-machine-learning/, 2024. Accessed on May 3, 2024.

[25] Jonathan Goh, Sridhar Adepu, Khurum Nazir Junejo, and Aditya Mathur. A dataset to support research in the design of secure water treatment systems. In Grigore Havarneanu, Roberto Setola, Hypatia Nassopoulos, and Stephen Wolthusen, editors, *Critical Information Infrastructures Security*, pages 88–99, Cham, 2017. Springer International Publishing. ISBN 978-3-319-71368-7.

[26] Clarke Gordon and Reynders Deon. *Practical Modern SCADA Protocols : DNP3, 60870.5 and Related Systems.* Practical Professional Books From Elsevier. Newnes, 2004. ISBN 9780750657990. URL https://search.ebscohost.com/login.aspx?direct=true&db=edsebk&AN=104779&site=eds-live&scope=site&authtype=guest&custid=s3911979&groupid=main&profile=eds.

[27] AKM Ahasan Habib, Mohammad Kamrul Hasan, Ahmed Alkhayyat, Shayla Islam, Rohit Sharma, and Lulwah M. Alkwai. False data injection attack in smart grid cyber physical system: Issues, challenges, and future direction. *Computers and Electrical Engineering*, 107:108638, 2023. ISSN 0045-7906. doi: https://doi.org/10.1016/j.compeleceng.2023.108638. URL https://www.sciencedirect.com/science/article/pii/S0045790623000630.

[28] Won-Seok Hwang, Jeong-Han Yun, Jonguk Kim, and Byung Gil Min. "do you know existing accuracy metrics overrate time-series anomaly detections?". In *Proceedings of the 37th ACM/SIGAPP Symposium on Applied Computing*, SAC '22, page 403–412, New York, NY, USA, 2022. Association for Computing Machinery. ISBN 9781450387132. doi: 10.1145/3477314.3507024. URL https://doi.org/10.1145/3477314.3507024.

[29] IBM. What is the internet of things (iot)? https://www.ibm.com/topics/internet-of-things, N.D. Accessed on April 10, 2024.

[30] Jason Brownlee. A gentle introduction to generative adversarial networks (gans). https://machinelearningmastery.com/what-are-generative-adversarial-networks-gans/, 2019. Accessed on April 15, 2024.

[31] Karel J. Keesman. *System Identification. [electronic resource] : An Introduction.* Advanced Textbooks in Control and Signal Processing. Springer London, 2011. ISBN 9780857295224. URL https://search.ebscohost.com/login.aspx?direct=true&db=cat07472a&AN=clec.SPRINGERLINK9780857295224&site=eds-live&scope=site&authtype=guest&custid=s3911979&groupid=main&profile=eds.

[32] Ken Proska, John Wolfram, Jared Wilson, Dan Black, Keith Lunden, Daniel Kapellmann Zafra, Nathan Brubaker, Tyler Mclellan, Chris Sistrunk. Sandworm disrupts power in ukraine using a novel attack against operational technology. https://www.mandiant.com/resources/blog/sandworm-disrupts-power-ukraine-operational-technology, 2023. Accessed on February 23, 2024.

[33] Aymen Rayane Khouas, Mohamed Reda Bouadjenek, Hakim Hacid, and Sunil Aryal. Training machine learning models at the edge: A survey. *arXiv preprint arXiv:2403.02619*, 2024.

[34] Ansam Khraisat, Iqbal Gondal, Peter Vamplew, and Joarder Kamruzzaman. Survey of intrusion detection systems: techniques, datasets and challenges. *Cybersecurity*, 2(1):20, Jul 2019. ISSN 2523-3246. doi: 10.1186/s42400-019-0038-7. URL https://doi.org/10.1186/s42400-019-0038-7.

[35] Jonguk Kim, Jeong-Han Yun, and Hyoung Chun Kim. Anomaly detection for industrial control systems using sequence-to-sequence neural networks, 2019.

[36] Kinza Yasar, Michael Cobb. Man-in-the-middle attack (mitm). https://www.techtarget.com/iotagenda/definition/man-in-the-middle-attack-MitM, 2022. Accessed on February 19, 2024.

[37] Kirkwood community collage. Real, reactive, and active power. https://kirkwood.pressbooks.pub/smartgrids/chapter/real-reactive-and-active-power/, N.D. Accessed on May 3, 2024.

[38] Moshe Kravchik and Asaf Shabtai. Efficient cyber attack detection in industrial control systems using lightweight neural networks and pca. *IEEE Transactions on Dependable and Secure Computing*, 19(4):2179–2197, 2022. doi: 10.1109/TDSC.2021.3050101.

[39] Olav Lamberts, Konrad Wolsing, Eric Wagner, Jan Pennekamp, Jan Bauer, Klaus Wehrle, and Martin Henze. [sok] evaluations in industrial intrusion detection research. *Journal of Systems Research*, 3(1), February 2023. ISSN 2770-5501. doi: 10.5070/sr33162445. URL http://dx.doi.org/10.5070/SR33162445.

[40] Olav Lamberts, Konrad Wolsing, Eric Wagner, Jan Pennekamp, Jan Bauer, Klaus Wehrle, and Martin Henze. [sok] evaluations in industrial intrusion detection research. *Journal of Systems Research*, 3(1), February 2023. ISSN 2770-5501. doi: 10.5070/sr33162445. URL http://dx.doi.org/10.5070/SR33162445.

[41] Dan Li, Dacheng Chen, Lei Shi, Baihong Jin, Jonathan Goh, and See-Kiong Ng. Mad-gan: Multivariate anomaly detection for time series data with generative adversarial networks, 2019.

[42] Qin Lin, Sridha Adepu, Sicco Verwer, and Aditya Mathur. Tabor: A graphical model-based approach for anomaly detection in industrial control systems. In *Proceedings of the 2018 on Asia Conference on Computer and Communications Security*, ASIACCS '18, page 525–536, New York, NY, USA, 2018. Association for Computing Machinery. ISBN 9781450355766. doi: 10.1145/3196494.3196546. URL https://doi.org/10.1145/3196494.3196546.

[43] Shuyu Lin, Ronald Clark, Robert Birke, Sandro Schönborn, Niki Trigoni, and Stephen Roberts. Anomaly detection for time series using vae-lstm hybrid model. In *ICASSP 2020 - 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 4322–4326, 2020. doi: 10.1109/ICASSP40776.2020.9053558.

[44] Yao Liu, Peng Ning, and Michael K Reiter. False data injection attacks against state estimation in electric power grids. *ACM Transactions on Information and System Security (TISSEC)*, 14(1):1–33, 2011.

[45] N. Loganathan, P. Mayurappriyan, and Kvn Lakshmi. Smart energy management systems: a literature review. *MATEC Web of Conferences*, 225:01016, 01 2018. doi: 10.1051/matecconf/201822501016.

[46] Juan Miguel López Velásquez, Sergio Mauricio Martínez Monterrubio, Luis Enrique Sánchez Crespo, and David Garcia Rosado. Systematic review of siem technology: Siem-sc birth. *International Journal of Information Security*, 22(3):691 – 711, 2023. ISSN 16155262. URL `https://search.ebscohost.com/login.aspx?direct=true&db=bsu&AN=163914439&site=eds-live&scope=site&authtype=guest&custid=s3911979&groupid=main&profile=eds`.

[47] Markus Kohler. Mqtt use cases: What is mqtt used for? `https://www.pubnub.com/blog/what-is-mqtt-use-cases/`, 2024. Accessed on Mars 26, 2024.

[48] Aditya P. Mathur and Nils Ole Tippenhauer. Swat: a water treatment testbed for research and training on ics security. In *2016 International Workshop on Cyber-physical Systems for Smart Water Networks (CySWater)*, pages 31–36, 2016. doi: 10.1109/CySWater.2016.7469060.

[49] P. Haji Mirzaee, M. Shojafar, H. Cruickshank, and R. Tafazolli. Smart grid security and privacy: From conventional to machine learning issues (threats and countermeasures). *IEEE Access, Access, IEEE*, 10:52922 – 52954, 2022. ISSN 2169-3536. URL `https://search.ebscohost.com/login.aspx?direct=true&db=edseee&AN=edseee.9772626&site=eds-live&scope=site&authtype=guest&custid=s3911979&groupid=main&profile=eds`.

[50] National Cyber Security Centre. How cyber attacks work. `https://www.ncsc.gov.uk/information/how-cyber-attacks-work`, 2015. Accessed on February 22, 2024.

[51] Man Ni. A review on machine learning methods for intrusion detection system. *Applied and Computational Engineering*, 27:57–64, 12 2023. doi: 10.54254/2755-2721/27/20230148.

[52] Dong Yul Oh and Il Dong Yun. Residual error based anomaly detection using auto-encoder in smd machine sound. *Sensors*, 18(5), 2018. ISSN 1424-8220. doi: 10.3390/s18051308. URL `https://www.mdpi.com/1424-8220/18/5/1308`.

[53] OWASP. OWASP Internet of Things Project. `https://wiki.owasp.org/index.php/OWASP_Internet_of_Things_Project`, 2018. Accessed: 2023-11-30.

[54] Victoria Pillitteri and Tanya Brewer. Guidelines for smart grid cybersecurity. `https://doi.org/10.6028/NIST.IR.7628r1`, 2014-09-25 2014.

[55] Ken Proska, John Wolfram, Jared Wilson, Dan Black, Keith Lunden, Daniel Kapellmann Zafra, Nathan Brubaker, Tyler Mclellan, and Chris Sistrunk. Sandworm Disrupts Power in Ukraine Using a Novel Attack Against Operational Technology. https://www.mandiant.com/resources/blog/sandworm-disrupts-power-ukraine-operational-technology, 2023. Accessed: 2023-11-30.

[56] Ruobin Qi, Craig Rasband, Jun Zheng, and Raul Longoria. Detecting cyber attacks in smart grids using semi-supervised anomaly detection and deep representation learning. *Information*, 12(8), 2021. ISSN 2078-2489. doi: 10.3390/info12080328. URL https://www.mdpi.com/2078-2489/12/8/328.

[57] Panagiotis Radoglou-Grammatikis, Panagiotis Sarigiannidis, Eider Iturbe, Erkuden Rios, Saturnino Martinez, Antonios Sarigiannidis, Georgios Eftathopoulos, Yannis Spyridis, Achilleas Sesis, Nikolaos Vakakis, Dimitrios Tzovaras, Emmanouil Kafetzakis, Ioannis Giannoulakis, Michalis Tzifas, Alkiviadis Giannakoulias, Michail Angelopoulos, and Francisco Ramos. Spear siem: A security information and event management system for the smart grid. *Computer Networks*, 193:108008, 2021. ISSN 1389-1286. doi: https://doi.org/10.1016/j.comnet.2021.108008. URL https://www.sciencedirect.com/science/article/pii/S1389128621001237.

[58] Panagiotis I. Radoglou-Grammatikis and Panagiotis G. Sarigiannidis. Securing the Smart Grid: A Comprehensive Compilation of Intrusion Detection and Prevention Systems. *IEEE Access*, 7:46595–46620, 2019. doi: 10.1109/ACCESS.2019.2909807.

[59] Raspberry Pi. Raspberry pi 400 tech specs. https://www.raspberrypi.com/products/raspberry-pi-400/specifications/, N.D. Accessed on May 7, 2024.

[60] Martin Roesch et al. Snort: Lightweight intrusion detection for networks. In *Lisa*, volume 99, pages 229–238, 1999.

[61] Indu Sekhar Samant, Subhasis Panda, and Pravat Kumar Rout. Recent advancements on cyber security for smart-grids: A survey. In *2023 International Conference in Advances in Power, Signal, and Information Technology (APSIT)*, pages 572–577, 2023. doi: 10.1109/APSIT58554.2023.10201710.

[62] Nike Sartika, Yuda Sukmana, Mufid Ridlo Effendi, Iu Rusliana, Khomisah Khomisah, and Yuyun Yuningsih. A Systematic Literature Review on IoT-based Smart Grid. In *2021 7th International Conference on Wireless and Telematics (ICWT)*, pages 1–5, 2021. doi: 10.1109/ICWT52862.2021.9678415.

[63] R. Sathya and Annamma Abraham. Comparison of supervised and unsupervised learning algorithms for pattern classification. *International Journal of Advanced Research in Artificial Intelligence*, 2, 02 2013. doi: 10.14569/IJARAI.2013.020206.

[64] William T. Shaw. *Cybersecurity for SCADA Systems.* PennWell Books, LLC, 2021. ISBN 9781593705053. URL `https://search.ebscohost.com/login.aspx?direct=true&db=cat07472a&AN=clec.EBC6468131&site=eds-live&scope=site&authtype=guest&custid=s3911979&groupid=main&profile=eds`.

[65] Abhijeet Singh, Achyut Mishra, Ajit Antil, Bharat Bhushan, and Anamika Chauhan. Anomaly based ids in industrial iot. *2023 International Conference on Smart Systems for applications in Electrical Sciences (ICSSES), Smart Systems for applications in Electrical Sciences (ICSSES), 2023 International Conference on*, pages 1 – 6, 2023. ISSN 979-8-3503-4729-6. URL `https://search.ebscohost.com/login.aspx?direct=true&db=edseee&AN=edseee.10199661&site=eds-live&scope=site&authtype=guest&custid=s3911979&groupid=main&profile=eds`.

[66] Boryana Stoytcheva, Maria Nenova, Zlatka Valkova-Jarvis, and Kiril Kassev. Security threats and models in the field of renewable energy systems. In *2023 Eight Junior Conference on Lighting (Lighting)*, pages 1–4, 2023. doi: 10.1109/Lighting59819.2023.10299494.

[67] Adarsh Madhukumar Vidya, Deeraj D. Sai, G. Sarveshwaran, S.S. Mukesh, and K. R. M. Vijaya Chandrakala. Identification of false data injection and man in the middle cyber-attacks impact on smart grid. In *2023 Third International Conference on Secure Cyber Computing and Communication (ICSCCC)*, pages 678–683, 2023. doi: 10.1109/ICSCCC58608.2023.10176782.

[68] xcitium. Living off the land (lotl) attacks. `https://www.xcitium.com/living-off-the-land-attacks/`, 2024. Accessed on February 23, 2024.

[69] Akshatha Y and A. S Poornima. Cyber-physical security system in smart grid-a review. In *2023 14th International Conference on Computing Communication and Networking Technologies (ICCCNT)*, pages 1–7, 2023. doi: 10.1109/ICCCNT56998.2023.10307396.

[70] Geeta Yadav and Kolin Paul. Architecture and security of scada systems: A review. *International Journal of Critical Infrastructure Protection*, 34, 2021. ISSN 1874-5482. URL `https://search.ebscohost.com/login.aspx?direct=true&db=edselp&AN=S1874548221000251&site=eds-live&scope=site&authtype=guest&custid=s3911979&groupid=main&profile=eds`.

[71] Jiuqi Elise Zhang, Di Wu, and Benoit Boulet. Time series anomaly detection for smart grids: A survey. *2021 IEEE Electrical Power and Energy Conference (EPEC), Electrical Power and Energy Conference (EPEC), 2021 IEEE*, pages 125 – 130, 2021. ISSN 978-1-6654-2928-3. URL `https://search.ebscohost.com/login.aspx?direct=true&db=edseee&AN=edseee.9621752&site=eds-live&scope=site&authtype=guest&custid=s3911979&groupid=main&profile=eds`.

[72] Kai Zhang, Fei Zhao, Shoushan Luo, Yang Xin, and Hongliang Zhu. An intrusion action-based ids alert correlation analysis and prediction framework. *IEEE Access*, 7:150540–150551, 2019. doi: 10.1109/ACCESS.2019.2946261.

A

Appendix 1

A.1 Attack prediction quality metric

The prediction quality metric was created by the authors and represents the overall quality of the predictions in all attack windows. More specifically, it calculates a percentage of the mean prediction score in the attack windows in the range between the detection threshold and the mean prediction value of non-attack windows. This metric effectively allows for quantifying the quality of the predictions produced by an algorithm even though an alarm is not triggered.

Firstly, the mean value of the predictions outside the attack windows is calculated.

Algorithm 1

```
non_attack_means ← []
prev_end ← 0
for each (index_start,index_end) in attack_windows do
    append predictions[prev_end:index_start] to non_attack
    prev_end ← end
end for
append predictions[prev_end:] to non_attack
non_attack ← flatten non_attack list
mean_score ← mean of non_attack
```

Then, the attack detection quality score is calculated as follows:

$$\max\left(0, \min\left(1, \frac{\max\left(1, \text{attack\_mean} - \text{non\_attack\_mean}\right)}{\max\left(1, \text{threshold} - \text{non\_attack\_mean}\right)}\right)\right) \tag{A.1}$$

A.2 SWaT evaluation

The SWaT evaluation aimed to verify that the algorithms functioned as intended, consistent with the specifications outlined in their respective papers. By using the SWaT dataset, which had previously been utilised to assess both algorithms, this process established a benchmark for their performance, with results available through various types of reports.

The results demonstrate that both PASAD and USAD performed as expected.

PASAD						
Training Points		496,800				
Lag		100	500	1,000	1,500	2,000
Detection Delay		35,566	7,234	5,797	5,075	2,328
Detection Accuracy		0.2857	0.4285	0.6571	0.7142	0.8
False Alarms		0	0	0	0	0
Precision		0.7722	0.1721	0.1648	0.1665	0.1506
Recall		0.6645	0.8444	0.8501	0.8724	0.8877
F1		**0.7143**	0.2860	0.2761	0.2796	0.2575
Point-Adjust	Precision	0.7913	0.1744	0.1700	0.1689	0.1524
	Recall	0.7436	0.8577	0.8821	0.8880	0.8999
	F1	**0.7667**	0.2898	0.2850	0.2839	0.2606
eTaPR	eTaP	0.8608	0.6169	0.6043	0.5839	0.5587
	eTaR	0.2461	0.5301	0.5658	0.6130	0.6448
	F1	0.3828	0.5702	0.5844	0.5981	**0.5987**
MAE		6.45e-2	5.12e-1	5.42e-1	5.47e-1	6.21e-1

Table A.1: SWaT evaluation using PASAD

USAD						
Training Points		496,800				
Window Size		8	12	20	30	50
Detection Delay		7,437	7,383	7,387	7,391	10,329
Detection Accuracy		0.2857	0.3142	0.3428	0.3428	0.2857
False Alarms		72	71	71	71	64
Precision		0.9154	0.9143	0.9114	0.9060	0.8972
Recall		0.6293	0.6279	0.6258	0.6229	0.6168
F1		**0.7459**	0.7445	0.7421	0.7382	0.7311
Point-Adjust	Precision	0.9256	0.9272	0.9247	0.9201	0.9104
	Recall	0.7238	0.7492	0.7470	0.7443	0.7178
	F1	0.8124	**0.8287**	0.8264	0.8229	0.8027
eTaPR	eTaP	0.4119	0.4232	0.4156	0.4083	0.4089
	eTaR	0.2452	0.2580	0.2586	0.2579	0.2439
	F1	0.3074	**0.3206**	0.3188	0.3161	0.3055
MAE		5.22e-2	5.26e-2	5.34e-2	5.46e-2	5.68e-2

Table A.2: SWaT evaluation using USAD

A.3 Desktop computer

The detailed specifications of the desktop computer used in the experiments is specified in Table A.3.

Hardware	Desktop computer
Processor	6-Core AMD Ryzen 5 5600X @ 3.70 GHz
Memory	32GB DDR4-3600
GPU	NVIDIA GeForce RTX 3060 Ti, 4864 CUDA cores
Storage	SSD

Table A.3: Specifications of the desktop computer used in the experiments