Intrusion Detection in SCADA Systems using Machine Learning Techniques

Leandros A. Maglaras

Supervisor: Prof. Andrew Crampton

Department of Computing and Informatics
University of Huddersfield

A thesis submitted to the University of Huddersfield
in partial fulfillment of the requirements for the degree of
Doctor of Philosophy

University of Huddersfield April 2018

I would like to dedicate this thesis to my loving family …

Declaration

Leandros A. Maglaras
April 2018

Acknowledgements

First of all, I want to thank the supervisor of my thesis, Professor Andrew Crampton, for his valuable help and guidance during my work. I am also grateful to the other members of the selection committee of my thesis, for careful reading of my work and for their valuable suggestions. I owe special thanks to Professor Jianmin Jiang, who helped throughout all my research and played a key role in the completion of my PhD. I also want to thank Ass. Professor T.J. Cruz whom we worked together in creating a cyber security framework that was tested on realistic scenarios in a Hybrid Testbed provided to us from IEC (Israel Electric Company).

Abstract

Modern Supervisory Control and Data Acquisition (SCADA) systems are essential for monitoring and managing electric power generation, transmission and distribution. In the age of the Internet of Things, SCADA has evolved into big, complex and distributed systems that are prone to conventional in addition to new threats. So as to detect intruders in a timely and efficient manner a real time detection mechanism, capable of dealing with a range of forms of attacks is highly salient. Such a mechanism has to be distributed, low cost, precise, reliable and secure, with a low communication overhead, thereby not interfering in the industrial system's operation.

In this commentary two distributed Intrusion Detection Systems (IDSs) which are able to detect attacks that occur in a SCADA system are proposed, both developed and evaluated for the purposes of the CockpitCI project. The CockpitCI project proposes an architecture based on real-time Perimeter Intrusion Detection System (PIDS), which provides the core cyber-analysis and detection capabilities, being responsible for continuously assessing and protecting the electronic security perimeter of each CI. Part of the PIDS that was developed for the purposes of the CockpitCI project, is the OCSVM module. During the duration of the project two novel OCSVM modules were developed and tested using datasets from a small-scale testbed that was created, providing the means to mimic a SCADA system operating both in normal conditions and under the influence of cyberattacks.

The first method, namely K-OCSVM, can distinguish real from false alarms using the OCSVM method with default values for parameters ν and σ combined with a recursive K-means clustering method. The K-OCSVM is very different from all similar methods that required pre-selection of parameters with the use of cross-validation or other methods that ensemble outcomes of one class classifiers. Building on the K-OCSVM and trying to cope with the high requirements that were imposed from the CockpitCi project, both in terms of accuracy and time overhead, a second method, namely IT-OCSVM is presented. IT-OCSVM method is capable of performing outlier detection with high accuracy and low overhead within a temporal window, adequate for the nature of SCADA systems.

The two presented methods are performing well under several attack scenarios. Having to balance between high accuracy, low false alarm rate, real time communication requirements

and low overhead, under complex and usually persistent attack situations, a combination of several techniques is needed. Despite the range of intrusion detection activities, it has been proven that half of these have human error at their core. An increased empirical and theoretical research into human aspects of cyber security based on the volumes of human error related incidents can enhance cyber security capabilities of modern systems. In order to strengthen the security of SCADA systems, another solution is to deliver defence in depth by layering security controls so as to reduce the risk to the assets being protected.

-

Table of contents

List of figures

List of tables

Glossary:

- ACK: ACKnowledge
- CI : Critical Infrastructure
- FIN : Finish
- ICS: Industrial Control System
- IDS : Intrusion Detection System
- OCSVM: One-Class SVM
- OSI: Open Systems Interconnection model
- PLC: Programmable Logic Controller
- RTU: Remote terminal unit
- RST: ReSeT
- SCADA : Supervisory Control and Data Acquisition
- SYN: SYNchronize

Chapter 1

Introduction

1.1 Aim and Objectives

The current commentary is presenting two novel Intrusion Detection Systems (IDS) that were produced from the candidate in 2014. These IDS were designed to be a part of a Perimeter IDS that was developed for the purposes of the CockpitCI project (see Section 1.5). Both methods were developed and implemented from the candidate solely, using datasets and live testbeds that were produced from partners of the consortium. The main objectives of the research that was conducted in 2014 and led to the creation of these two novel methods, were the creation of an IDS that could cope with novel attacks, operate in real time, be accurate, could be easily integrated with the rest of the detection components, be distributed and induce low communication overhead. Initially the K-OCSVM method was developed, but the strict aforementioned requirements especially about high accuracy, led to the creation of the second method, namely IT-OCSVM that met all the objectives.

1.2 Industrial Control Systems

An Industrial Control System (ICS) is an umbrella term that refers to a group of process automation technologies, such as Supervisory Control and Data Acquisition (SCADA) systems and Distributed Control Systems (DCS). Unfortunately, they have been subject to a growing number of attacks in recent years [1]. As they deliver vital services to critical infrastructure, such as communications, manufacturing and energy among others, hostile intruders mounting attacks represent a serious threat to the day to day running of nation states [2, 3].

ICS have unique performance and reliability requirements and often use operating systems, applications and procedures that may be considered unconventional by contemporary IT professionals. These requirements typically follow the priority of availability and integrity, followed by confidentiality and include the management of processes that, if not executed correctly, pose a significant risk to the health and safety of human lives, damage to the environment, as well as serious financial issues such as production losses [4–6]. Failure or security breaches of these systems could result in wide-reaching adverse impacts, not only for the system under attack, but for the community and economy at-large that depends on goods or services of the organization. For example, wide-scale blackouts due to a failure of the electricity grid or an environmental disaster such as an oil or sewage pipeline releasing its contents, plus any collateral impact. Cyber attacks could contribute towards the collapse of a state if they initiate or prolong the failure of critical national infrastructure [7, 8].

Cyber warfare which causes blackouts, cuts off supplies to safe drinking water, makes travelling dangerous or destabilises a national economy is clearly a threat to the stability of a nation and is therefore a threat to international peace and security [9]. Unavailability of critical infrastructure (e.g., electrical power, transportation) can have economic impact far beyond the systems sustaining direct and physical damage. These effects could negatively impact the local, regional, national, or possibly global economy [4]. When an incident occurs on a CI, the consequences can be catastrophic having an impact on the environment, economy and human life and a forensic investigation needs to take place [10, 11]. Taking one step further, Robinson et al. in [12] cyber warfare will threaten civilian peace and security long after a conflict has ended, and that existing peace operations will be required to evolve in order to address this threat.

Despite the apparent risk to critical infrastructure, the security of ICS is not considered a significant investment area. Naedele et al., in [13] argues that the costs involved in ICS security are prohibitive, especially within critical systems, when the perceived risks to an organisation or infrastructure cannot be adequately quantified and a business case not satisfactorily articulated. This often leads to an underdeveloped incident response capability in the deployed operational ICS in particular within the SME supply chain. Larger infrastructures suffer from the insufficient understanding of the deployed components such as Programmable Logic Controllers (PLC) or similar Intelligent Electronic Devices (IED), Remote Terminal Units (RTU) and input/output (I/O) devices that are used to manage electromechanical equipment in either local or distributed environments. This unique environment, that combines large scale, geographically distributed, legacy and proprietary system components presents significant challenges to Security Operation Centers (SOCs) and Cyber Emergency Response Teams (CERTs)[14]. A SOC centralizes the roles responsible for protecting information

security in the organization, and includes prevention, detection, incident management and response, reporting, governance, risk, and compliance. A CERT is a centralized function for information security incident management and response in an organization. It may roll up under a SOC, or it may act as the main security organization depending on your company's structure and security needs.

In the past, ICS were operated as separated networks unconnected to public communication infrastructures [15], but as businesses have turned to exploiting the services and data provided by the Internet, such isolation that protected these systems has declined [16]. The benefits afforded by real time monitoring, peer to peer communications, multiple sessions, concurrency, maintenance and redundancy have enhanced the services provided for consumers and operators [17, 15]. Moreover, this interconnectedness will grow with the implementation of smart grids and execution of the Internet of Things (IoT) [18]. Hence, the previously isolated systems have become increasingly exposed to a range of threats [19, 20]. Regarding which, Byres et al,[21] cite that formerly isolated ICS now average 11 direct connections across networks with weak network segmentation.

IT security is generally focused on protecting networked computer assets with clear, shared attributes, but Zhu [22] argues that for securing ICS there needs to be a combination of conventional computer security and communication networking with control engineering. However, because of the of the current ICS systems, and their only having recently taken up IPbased communications and linked devices, where traditional IT security, communications security and the protection of control systems have their boundaries, remains unclear and hence, their efficacy is still to be proven [20, 23]. Luallen [24] reports that for a survey of 268 respondent organisations to a survey, most did not report critical ICS assets and relied on staff to detect issues, not tools.

1.3 SCADA Systems

SCADA systems have traditionally been associated with a subset of ICS referred to as Wide Area Control systems (see Figure 1.1).

As aforementioned, security in SCADA systems is more salient than with most other computer systems owing to the potential severity of the outcomes due to a degrading of service as well as the disruption to day to day life. With older computer systems, reliability was the key concern and security was much further down the list. Today, with greater connectivity [25], security is now high on the agenda. Moreover, SCADA systems are not only becoming more connected to the internet; the communications within them operate

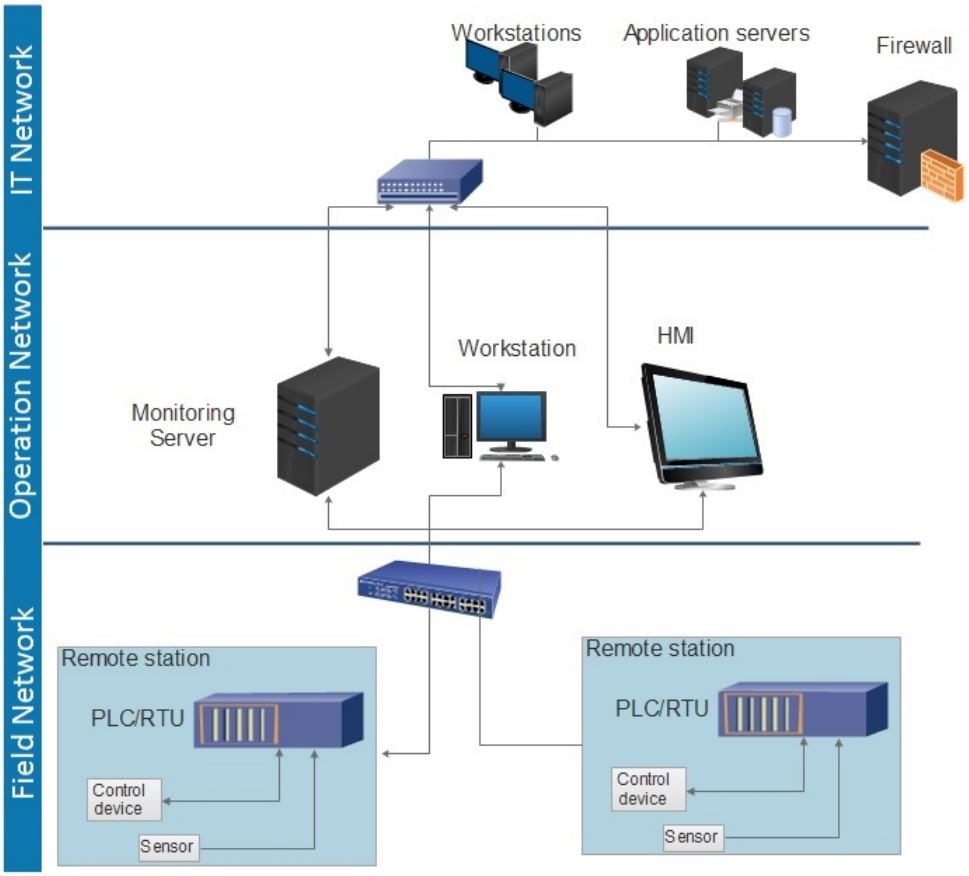

Fig. 1.1 A typical SCADA system

through shared Internet Protocol (IP) infrastructure [26]. A number of concerns in relation to implementing security in SCADA have been raised in the extant research:

- System reliability regularly takes precedence over threats to security and can result in high security vulnerability.
- Absence of encryption in earlier communication protocols (plain text is often utilised).
- The common used well-documented protocols, default passwords and off the shelf hardware solutions can threaten to undermine obscurity [27]. Whilst this is not a mechanism of security per se, the loss of it can lead to attacks becoming easier.
- The operation of an SCADA has to be ongoing, which makes it very hard to apply updates, perform patching or to modify system components.
- Today's systems are lasting longer than in the past, which means that hardware and software are operating beyond their supported lifespan [28]. Also many experienced

workers possess critical know-how that, if lost, makes it costly—if not impossible—for the company to replace when retired or moved.

The aforementioned specific characteristics and constraints in relation to SCADA mean that a domain-specific approach is necessary. In-line security mechanisms (e.g. traditional network IDS utilization) or security tools at the host level (e.g. anti-virus) are not recommended owing to possible latency impact or the occurrence of single points of failure along the vital communications path. Further, given the increasing sophistication of attacks, cyber-security no longer can depend on supervised, pattern-based detection algorithms to guarantee continuous security monitoring. There needs to be approaches that handle rogue threats, which provide a suitable balance between maintenance and detection power [23].

1.4 Real-world Attacks

Among the others, the STUXNET worm infection [29] perfectly represents the frailty of the regulatory systems devoted to control critical infrastructures. Although neither country has openly admitted responsibility, the worm is believed to be a jointly built American/Israeli cyberweapon First isolated in mid-June 2010, STUXNET was a computer virus specifically designed for attacking Windows based industrial computers and taking control of Programmable Logic Controller (PLCs), influencing the behaviour of remote actuators and leading to instability phenomena or even worse. The paradox is that critical infrastructures massively rely on newest interconnected (and vulnerable) Information and Communication Technology (ICT) technologies, while the control equipment is typically old, legacy software/hardware. Such a combination of factors may lead to very dangerous situations, exposing the systems to a wide variety of attacks. The lesson the CIIP (Critical Information Infrastructure Protection) community has learned from the spread of the STUXNET worm is that, in order to effectively react to a specific low level menace, there is the need to consider both the global and the local perspectives. In fact, besides obtaining a wider perspective on the state of the System of Systems, there is the need to increase the intelligence of equipments and devices that are used to influence the behaviour of the system, such as RTUs, valves, etc. Taking in mind that such worms can be launched from national intelligence agencies with vast resources for performing persistent attacks [30], only solutions that combine several methods can have adequate detection and mitigation performance.

Moreover, as emphasised by several episodes [31], another effective way to paralyse a SCADA system via cyber attack is to saturate the bandwidth of the carrier used for the communication (this was, for example, the way in which the SLAMMER worm operated in 2003 to affect the SCADA of two United States (US) utilities and a nuclear power plant).

Indeed, as emphasised also by the ANSI/ISA.99 (American National Standards Institute/ International Society of Automation), availability is the most crucial attribute of information security. The lack of timely information to/from the field may cause dramatic consequences because the field is unable to receive the adequate command, hence even trivial episodes may provoke dramatic impact, as shown by the US black-out.

In an evaluation of the Mariposa botnet infection in an ICS organisation, the US Department of Homeland Security [32, 33] explained that they found that the infection occurred when an employee used a USB drive to download presentation materials to a corporate laptop. When the user connected the laptop to the corporate network upon returning to work, the virus spread to over 100 hosts.

The security of SCADA communications is becoming more complicated because the decision has been taken to link the SCADA networks with IT networks to allow better and faster communications. But these new features have increased the threats and risks on SCADA communications. There are presently no convinced solutions to enforce the security of SCADA communications in that perspective. The idea to add "intelligence" to the field is not new; electro-valves for gas pipelines are available on the market that, in the case they receive a rapid sequence of open-close commands, do not perform them in order to avoid the consequence of the mechanical shock and EU (European Union) projects such as the FP6 SAFEGUARD and FP7 CRUTIAL (CRitical UTility InfrastructurAL Resilience) and the SCADA-CSL project have explored the technical feasibility to improve cyber security of SCADA system by improving the smartness of the field devices. However, until now such solutions have been completely rejected by Critical Infrastructure (CI) operators because they fear that local automatic reactions may happen during "normal" activities inducing catastrophic behaviour.

To overcome such catastrophic vision, and in order to improve the resilience and the dependability of CIs solutions, like the design and the implementation of alerting Systems that provide CI operators an efficient tool to support them, must be designed. Such tool could provide the operators with capabilities to assist them: (i) in the prevention of cyber attacks, and (ii) in the implementation of consequence containment strategies in case of an attack.

1.5 CockpitCi Project

The CockpitCI project proposes an architecture based on real-time Perimeter Intrusion Detection System (PIDS), which provides the core cyber-analysis and detection capabilities, being responsible for continuously assessing and protecting the electronic security perimeter of each CI. The analysis and detection mechanisms for each PIDS are fed by several field

adapters and detection agents, deployed within each CI and providing the basic information from which the ongoing security status of the CI is inferred. The PIDS also encompasses semi-autonomous reaction capabilities, being able to deploy countermeasures, in line with predefined security reaction policies.

The CockpitCI PIDS (see Figure 1.2) is structured along the three different zones of the CI, each one with its own internal security perimeter: the Field Network, SCADA Process/Operations Network and the IT (Information Technology) Network. This distinction confers the PIDS the ability to deploy agents and security policies customized to the specific needs and characteristics of each network scope.

The architecture components are arranged in a two-level architecture with local instances fine-tuned for each network scope: local correlators perform the first step of correlation, receiving the events from local detection agents (e.g. Host IDS) on their network scope and process them accordingly with a set of rules, forwarding significant results to a global correlation engine. After local correlation, events are sent to the global correlators, which have access to a broad perspective of the whole infrastructure, covering all network scopes. This approach provides context separation, also allowing for better efficiency and scalability for (soft)real time event processing. This architecture was designed to deal with several attack scenarios, from known threats to rogue events, such as: man-in-the-middle attacks, device impersonation, non-authorized tampering, worms, trojans, denial-of-service attacks or flooding, among others. For this purpose, the PIDS is designed in such a way that it integrates different detection strategies, distributed along different levels, namely:

- Detection agents and field adaptors, including agents, adaptors and extensions for existing system components, as well as specialized probes and honeypots to be added to the network which are able to capture behaviour or traffic patterns (as performed by NIDS – Network IDS components) as well as host (using Host IDS, or antivirus software) and field device monitoring.

- A distributed multi-zone, multi-level rule-based correlation structure that processes the information provided by the security sensors, complemented by machine-learning capabilities, in the form of One-Class Support Vector Machines (OCSVM) anomaly detection module, based on machine learning.

- Aggressive usage of topology and system-specific detection mechanisms, based on the fact that the role and behaviour of each system component in an ICS are expected to be more consistent over time than on other types of networks, analysis components are fed with knowledge provided by a number of system specific sources, such as topology

databases, policy databases, and trust-based mechanisms, as well as strategically placed honeypots.

In the CockpitCI PIDS, computational intelligence is provided by means of the analysis components, which provide a way to extract information from the data collected by the agent layer or directly from network traces.

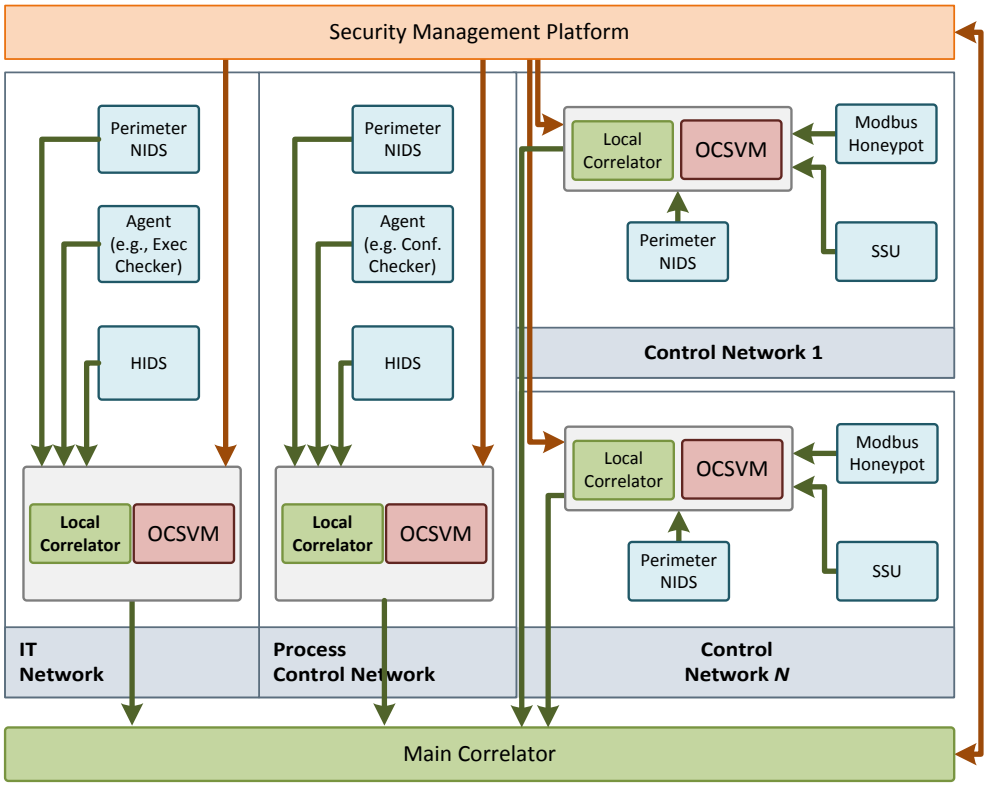

Fig. 1.2 High-level view of the CockpitCI PIDS architecture

Part of the PIDS that was developed for the purposes of the CockpitCI project, is the OCSVM module. During the duration of the project two novel OCSVM modules were developed and tested using datasets from a small-scale testbed that was created, providing the means to mimic a SCADA system operating both in normal conditions and under the influence of cyberattacks. Initial tests were conducted in order to evaluate the performance of the IT-OCSVM mechanism using three types of attacks, namely: network scan, Man-in-the-Middle (MITM) using ARP spoofing and Denial-of-Service (DoS). The Hybrid Environment for Development and Validation (HEDVa) that was designed by the Israeli Electric Company for the development and validation of CIs as well as security research projects, was used in a later stage in order to implement the attack use cases for the PIDS validation effort.

The following chapters describe in detail the two proposed IDS systems that are based on the OCSVM mechanism, namely the IT-OCSVM and the K-OCSVM. Both systems that are novel were developed by the candidate under the supervision of Professor Jiang during his work as research fellow in University of Surrey in 2014. Initially the K-OCSVM method was developed, but requirements about fast reaction of the IDS along with high accuracy, led to the creation of the second method, namely IT-OCSVM. This new method performed well both in terms of time and accuracy, was easily integrated in the PIDS and attracted the attention of the research community (Interview of the candidate by IET Electronics Letters (http://digital-library.theiet.org/content/journals/10.1049/el.2014.4062), keynote speech to ICAIDM 2014 conference, invited talk to CN group of TU WIEN in 2017 etc).

Chapter 2

Related Work

SCADA systems that link facilities like oil and gas pipelines or water distribution and waste water collection systems might have been designed to be open, robust, and easily operated and repaired, but they are not necessarily secure. Cyber-attacks against these systems are considered extremely dangerous for CI (Critical Infrastructures) operation and must be addressed in a specific way [22]. Intrusion Detection Systems can be used in order to identify an attack and initiate proper alerts that may help take appropriate actions.

Artificial Intelligence, machine learning and data mining are commnly used in every day applications. From an intrusion detection perspective, analysts can apply machine learning, data mining and pattern recognition algorithms to distinguish between normal and malicious traffic [34]. Artificial Intelligence (AI) is about improving algorithms by employing novel problem solving techniques. AI systems integrate several tasks such as learning or gaining the ability to perform tasks from simple or complex examples and extensive training. Classifier systems attempt to learn how to classify future examples from a set of training data. Neural networks are characterized by highly connected networks which are trained on a set of data counting on the hypothesis that the network will correctly classify future examples. In literature, numbers of anomaly detection systems are developed based on many different machine learning techniques. For example, some studies use single learning techniques, such as neural networks, genetic algorithms, support vector machines, etc. As the amount of data that need to be monitored and analyzed, in real time, is huge, data mining techniques are used. Data mining is the process of discovering patterns in large data sets involving methods at the intersection of machine learning and artificial intelligence. There is a significant overlap between machine learning and data mining. These two terms are commonly confused because they often employ the same methods although the former focuses on classification and prediction and the latter on the discovery of previously unknown patters [35]. Modern

intrusion detection systems that can cope with zero days attacks in an efficient and accurate way rely on artificial intelligence, machine learning and data mining techniques.

2.1 Intrusion Detection Systems

Intrusion refers to actions geared towards challenging the integrity, discretion or accessibility of a resource [36]. Whereas intrusion protection pertains to the monitoring of events that occur in a computer system and analysing them so as to detect any possible intrusion [37]. An intrusion detection system (IDS) refers to the operation of intrusion detection principles and mechanisms across a network. It can have both software and hardware components running on host machines that track user activities. That is, it can have programs searching for potential insider threats to the host as well as distributed agents, that by scrutinizing network traffic, can detect threats from outsiders. The purpose of an *IDS* is to warn administrators about any suspect activity and in sometimes it even deals with an attack. In comparison to other types of security, e.g. firewalls, control of access or encryption, for which the purpose is one of securing the computer system, IDSs are tasked with detecting attackers that have already entered its internal network. Hence, it is strongly advised that the intrusion prevention and intrusion detection security practices be utilised in tandem, as this will enhance system security.

In the past, administrators manually controlled intrusion detectors (IDs). They monitored every activity on a console and identified inconsistencies. However, this was not effective and resulted in a high volume of errors, particularly as the amount of network activity steadily increased. Log file readers were developed to address this, which allowed for rapid detection of irregularities and unauthorised access. Audit log IDs introduced during the initial stages were a type of forensics. Whilst ID was post analysis and analysis of intrusions and changes to the structure of the system were detected some time after the event. This involved a long-winded process, being subject to error, particularly when system usage was heavy [38].

Work was carried out in the 1980s and 1990s to make IDSs more robust, with a solution being provided by Denning [39]. They created an IDS that could review audit data in real time, which thus was the first time that attacks could be pre-empted [37]. With the coming of the technological age, the IT security market expanded rapidly and IDSs were developed further and became widely available. Features, such as new alert modes, updating to attack pattern definitions, dedicated user friendly interfaces and methods that stopped identified attacks immediately were added. As various new attacks were generated across the web, such as the millennium bug, it soon became obvious that continuous innovation of security systems would be needed to protect computing networks.

There are two principle categories of IDSs and a compound one: anomaly, signature and hybrid/compound detection systems [40]. An IDS works using models to establish whether a series of actions can be classed in terms of good or bad and thus potentially be an intrusion. Anomaly detection systems are aimed at understanding the behaviour of the network traffic, whilst misuse detection systems compare the present situation with recognised misbehaviour patterns.

2.1.1 Anomaly Detection Systems

The decision making process in these systems is determined by taking into account the system or user's normal profile. The detector establishes normal behaviour for the observed object of interest (application, user, resource usage etc.), subsequently flagging up the proportion of suspicious activity and taking appropriate action [41]. Because no previous knowledge of the intrusion is required this type of detector is effective for unknown attacks [42]. Nevertheless, it has a high rate of false positives and is incapable of providing details about attacks [43].

Lemaire et al [44] point out that, traditionally, intrusion data is inspected by human analysts who evaluate the alerts and take decisions accordingly. Machine learning, it is argued, has the capability to gather knowledge about new data and make predictions about the new data based on the knowledge of previous data, leading to a far more efficient self-learning process. This sentiment is echoed by Garitano [45] and Larkin et al [20] who note that whilst anomaly-based IDSs are not usually used on traditional IT networks because of their dynamic environment, the predictable traffic patterns of an ICS lend themselves to such an approach. However, this concept is based on the premise that ICS traffic is, indeed, predictable.

2.1.2 Misuse Detection

These systems, also called signature based detection systems, utilise patterns of known attacks or critical points so as to identify and match recognised intrusions. They provide much greater accuracy, also being able to identify variants of known attacks. Moreover, this type of IDS can provide useful diagnostics about an attack in terms of why an alarm has been triggered [46]. Nonetheless, there are some drawbacks, such as lacking the capacity to identify novel intrusions. Recently, a combination of misuse and anomaly systems have been deployed to improve the rates of both recognised and unrecognised intrusions as well as to lower the rate of false positives [47]. However, for the sake of convenience, most commercial systems only deploy misuse detection systems.

2.1.3 Clustering

Clustering refers to unsupervised learning algorithms which do not require pre-labeled data to extract rules for grouping similar data instances. When using clustering scholars follow one of the three following assumptions:

1. **Purity assumption**: we can create clusters of only normal data, and thus any data that don't fit into one cluster are considered anomalous. This assumption is only valid when dealing with testbed scenarios and cannot be applied in real environments.

2. **Robustness assumption**: we can create clusters that contain both normal and anomalous data, but in this occasion using distance metrics we can spot anomalous data since they lie far away from the centroids [48]. This assumption produces false positives especially in attack free scenarios.

3. **Frequency assumption**: we create clusters of various sizes and the small ones contain anomalous data since attacks are not the rule but rather the exception. This assumption is not valid when dealing with DoS or DDoS attacks when the volume of malicious packets is high.

Using all these assumptions and by calibrating several parameters and thresholds, scholars have managed to develop IDSs that are based on clustering techniques and that can perform well under several attack scenarios [49–51]. On the other hand such assumptions make these methods inappropriate for producing high accurate alarms, when dealing with several attacks simultaneously and when having to deal with high volumes of data in real time [52]. Exploiting the basic characteristic of clustering that is to group together similar things and having as one of the basic objectives to create and IDS that doesn't overwhelm the medium with alarm messages, in the core of the two IDSs that are presented in this thesis lies the K-means method.

2.1.4 Data Mining for Intrusion Detection

The increasing deployment of data mining approaches has resulted in the creation of a whole host of algorithms for database analysis, machine learning and pattern identification, all of which can be utilised as an IDS. For instance, there is a range of algorithms available for mining audit data. To construct a model, these algorithms have to begin with analyis of the available data so as to identify any patterns or trends. The appropriate models are then created with the optimal parameters identified from the outcomes of the initial analysis The process of implementing an IDS through data mining is as follows: data collection, data

selection, preprocessing, transformation, data mining, evaluation and reporting, all of which were carried in the two proposed methods that are presented and analyzed in the following Chapters.

2.1.5 Support Vector Machines

SVMs utilise a method for classing data rapidly. First, training data are used to develop a model, which is then utilised to categorise new data. SVMs employ a kernel function for mapping data onto a higher dimensional feature space, whereby hyperplanes can be used to separate the different categories. Iteration is deployed to identify the best hyperplane for category distinction, which has the greatest margin between it and the data points that are nearest. The data points coinciding with this margin are named the support vectors.

Whilst it is feasible to have multiple classes for data allocation, the interest here lies in One-Class SVMS (OCSVM), which simply decides whether new data belongs to a single class or not. This gives it the capability of identifying previously unknown, suspicious behaviour and, rather than categorising it, just recognises it as not being good. Given the lack of any data points from another category, the approach takes the origin as the class 1 perfect vector and the optimal distance is between the origin and where the data point has left the category. It is essential to elicit appropriate values so as to obtain optimal outcomes with this model. This process is frequently improved by the use of other machine learning techniques to determine optimum values for the algorithm that can be obtained by utilising machine learning methods [53, 54].

2.1.6 Discussion

There has been a considerable amount of work undertaken regarding SCADA intrusion and anomaly detection, including Intrusion Detection Systems (IDSs) [55], or Distributed IDSs [56, 57, 57] device-level anomaly detection [58, 59] and classification [60], IDSs solutions combining network traces and physical process control data [61], detection based on traffic and protocol models [62] [63–65] and approaches based on machine-learning techniques [66, 67], just to mention a few. However, not all solutions are equally feasible: for instance, device-level anomaly detection requires adding hardware mechanisms for probing. Also, detecting attacks targeting loss of process visibility, such as Man-in-the-Middle (MITM) or process-level attacks (which attempt to disturb the SCADA process) is out of scope for several of these proposals.

Among other approaches, neural networks, support vector machines, K-nearest neighbor (KNN) and the hidden Markov model can be used for intrusion detection. While existing

Table 2.1 Differences between IT and CPS Intrusion Detection [6]

An IT IDS monitors host or network-level activity (e.g., an HTTP request or a web server	A CPS IDS monitors the physical processes (and hence the laws of physics) that govern behaviour of physical devices that make certain behaviours more likely to be seen than others.
An IT IDS monitors user-triggered activities, leading to unacceptably high false-positive rates due to the unpredictability of user behaviours.	A CPS IDS monitors activities that are frequently automated and time driven in a closed-loop setting, thus providing some regularity and predictability for behaviour monitoring
An IT IDS deals with mostly non-zero-day attacks rendering knowledge-based detection effective.	A CPS IDS deals with zero-day or highly sophisticated attacks, rendering knowledge-based detection ineffective.
An IT IDS often does not have to deal with legacy components, making behaviour specification of the physical processes governing legacy components unnecessary.	A CPS IDS often must deal with legacy technology, making behaviour-specification detection an effective technique by precisely specifying the physical processes governing the behaviour of legacy components.

signature-based network IDSs, such as Snort or Suricata can be effective in SCADA environments, they require specific customization for such a purpose [68, 69]. Also, they are not effective against rogue threats for which known patterns or signatures are not known. OCSVM principles have shown great potential in the area of anomaly detection [70], being a natural extension of the support vector algorithm in the case of unlabeled data, especially for the detection of outliers. As stated in [71] there is a need for developing a robust, and preferably real-time IDS technique that can work in unsupervised mode and have a high detection accuracy.

Mitchell et al. in in [6] and Cook et al. in [72] explore the viability of traditional IT security mechanisms in Cyber Physical Systems, such as ICS, and considers their effectiveness in supporting ICS cybersecurity incident response. Based on the findings in [6] it is obvious that IDSs that are designed for CPS systems differ compared to those that are developed for traditional IT (See Table 2.1).

The single-class categorisation is a particular case of a two class classification problem, whereby data from onle one class are present, known as the target class. A second category, called the outlier class, is sampled very sparingly, or maybe not at all. The data here indicate abnormal behaviour and hence, the possibility of an intrusion; OCSVM [66] has a number of positive attributes when processing data in a SCADA environment. The one-class

Support Vector Machine (OCSVM) [73] possesses several advantages for processing SCADA environment data and automates SCADA performance monitoring, which can be highlighted as follows:

- Regarding monitoring of SCADA performance the difference between what is normal and what is not might not be clear to the operators. Since OCSVM needs no data signatures, data for constructing the detection model it is very effective for identifying intrusion in SCADA environments.
- Because no previous information regarding anticipated types of attack is required, OCSVM can identify both recognised and unrecognised (new) intrusions.
- In reality, training data obtained in a SCADA environment can contain samples of noise, which most attack detection approaches are extremely sensitive to. OCSVM is robust to these samples during the process of training.
- The user can control the configuration of the algorithm in relation to the proportion of anticipated anomalies.
- OCSVM detectors are quick enough to perform online with SCADA performance monitoring, owing to the low computation time.
- Typical monitoring of data of SCADA systems consists of several attributes and OCSVM is capable of handling multiple-attributed data.

Based on this analysis and the need for an IDS that can be accurate, deal with novel attacks and work in real-time two novel methods that were based on the OCSVM mechanism were developed. In real-time systems, in addition to fast response and accuracy, limited communication between detection modules is also desirable. By sending an explicit message for every anomaly detected, the intrusion detection mechanism will flood the medium with messages that will cause a delay in the communication between entities in the SCADA system. Moreover, since the detection mechanism needs to be sited at several locations in the SCADA system in order to recognize the intrusion near the origin, the communication overhead caused by the detection mechanism is further increased. One solution is the addition of a control channel, whereby these messages can be communicated without affecting the system's performance, but this is not always feasible. For this reason, the IT-OCSVM mechanism that was the second and more successful solution presented, includes an aggregation mechanism that groups initial alerts and sends a limited number of messages reporting the fault/intrusion accurately and on time. Both IT-OCSVM and K-OCSVM, that was the initial mechanism developed, are presented in the following chapters. Both mechanisms were developed in

order to work real-time in collaboration with other detection agents, as a part of a Perimeter IDS.

This commentary presents two novel IDSs systems that were developed under the scope of a PIDS. These mechanisms were developed in order to overcome the shortcomings of all current (back in 2014) methods, in terms of accuracy, overhead and efficient countering of zero days attacks. The IT-OCSVM method, which is the improved and final IDS method produced was evaluated under realistic scenarios and was found to be very efficient in terms of time overhead and accuracy under all kinds of attacks, making it a central part of the PIDS. In Appendix A, the articles that describe the methodology, the evaluation and the main findings of the two aforementioned methods along with the integration of the later into the PIDS that was implemented as part of the CockpitCI project are listed. The current commentary presents both methods, the need that led to the design and implementation of these novel techniques and the evaluation of later under realistic attack scenarios.

Chapter 3

Proposed Methods

3.1 K-OCSVM Idea

OCSVM principles have shown great potential in the area of anomaly detection [70, 74, 75]. IDS can provide active detection and automated responses during intrusions [76]. Several extensions of the OCSVM method have been introduced lately [77, 78]. OCSVM, similar to other one-class classifiers e.g. GDE[79], PGA [80], suffer from false positive and over fitting situations. Intrusion detection systems (IDS) fail to deal with all kinds of attacks, while on the other hand, false alarms that are arisen from high sensitive IDS lead to high economic risks. A big number of false alarms fired from the IDS gives the impression that IDS technology itself is fundamentally flawed and that there is no possibility for a true false positive to exist. That way the IDS is ignored from the operator and the system is vulnerable to real attacks.For example when deploying a network based IDS, changes made to system files from an unauthorized user who uses the system console, goes unnoticed.

For the OCSVM with an Radial Basis Function (RBF) kernel, two parameters σ and ν need to be carefully selected in order to obtain the optimal classification result. Parameter ν sets an upper bound on the fraction of outliers (training examples regarded out-of-class) and, it is also a lower bound on the number of training examples used as Support Vectors. Parameter σ denotes the width of the Gaussian kernel (how tight the boundary fits over the training data). OCSVM, similar to other one-class classifiers suffer from false positives and over fitting. The former is a situation that occurs when the classifier fires an alarm in the absence of any real anomaly in the system and happens when the parameter σ has too large value. The latter is the situation when a model begins to memorize training data rather than learning to generalize from the trend and it shows up when the parameter σ is given a relatively small value [81].

A common strategy is to separate the data set into two parts, of which one is considered unknown. The prediction accuracy obtained from the unknown set more precisely reflects the performance on classifying an independent data set. An improved version of this procedure is known as cross-validation. Cross-validation is a model validation technique for assessing how the results of a statistical analysis will generalize to an independent data set. It is mainly used in settings where the goal is prediction, and one wants to estimate how accurately a predictive model will perform in practice.

In k-fold cross-validation [82, 83], the training set is divided into k subsets of equal size. Sequentially one subset is tested using the classifier trained on the remaining k .. 1 subsets . Thus, each instance of the whole training set is predicted once so the cross-validation accuracy is the percentage of data which are correctly classified. The cross-validation procedure can prevent the over fitting problem.

Using an ensemble of decision mechanisms with different parameters, is another method to have an optimal result. An ensemble of classifiers [84] is a set of classifiers whose individual decisions are combined in some way, in order to provide more trusted final decisions. Ensemble systems of classifiers are widely used for intrusion detection in networks. Classifier ensemble design aims to include mutually complementary individual classifiers which are characterized by high diversity, either in terms of classifier structure [85], internal parameters [86] or classifier inputs [87].

Unnthorsson et al. [88] proposed another method to select parameters for the OCSVM. In their method, ν was first set to a user-specified allowable fraction of misclassification of the target class (e.g. 1% or 5%). With ν fixed, σ is adjusted until a classification accuracy of $1 - \nu$ is obtained. The obtained ν and σ combination can then be used in the OCSVM classification.

In this subsection we present a novel method that is based on the combination of the OCSVM method with recursive K-means clustering separating the real from false alarms in real time and with no pre-selection of parameters σ and ν. The proposed method is a natural extension of the support vector algorithm to the case of unlabeled data, especially for the detection of outliers. The novel K-OCSVM mechanism is trained offline by network traces, after the attributes are extracted from the network dataset. Output of the detection module is communicated to the system by IDMEF files that contain information about the source, time and severity of the intrusion. After the execution of the K-OCSVM method, only severe alerts are communicated to the system by IDMEF files that contain information about the source, destination, protocol and time of the intrusion. The main feature of the K-OCSVM module is that it can perform anomaly detection in a time-efficient way, with good accuracy

and low overhead. A detailed description and evaluation of the K-OSCVM method can be found in [67, 89, 90].

3.2 K-OCSVM Implementation

The proposed K-OCSVM combines the well known OCSVM classifier with the RBF kernel with a recursive K-means clustering module. Figure 3.1 illustrates the procedure of intrusion detection of our proposed K-OCSVM model.

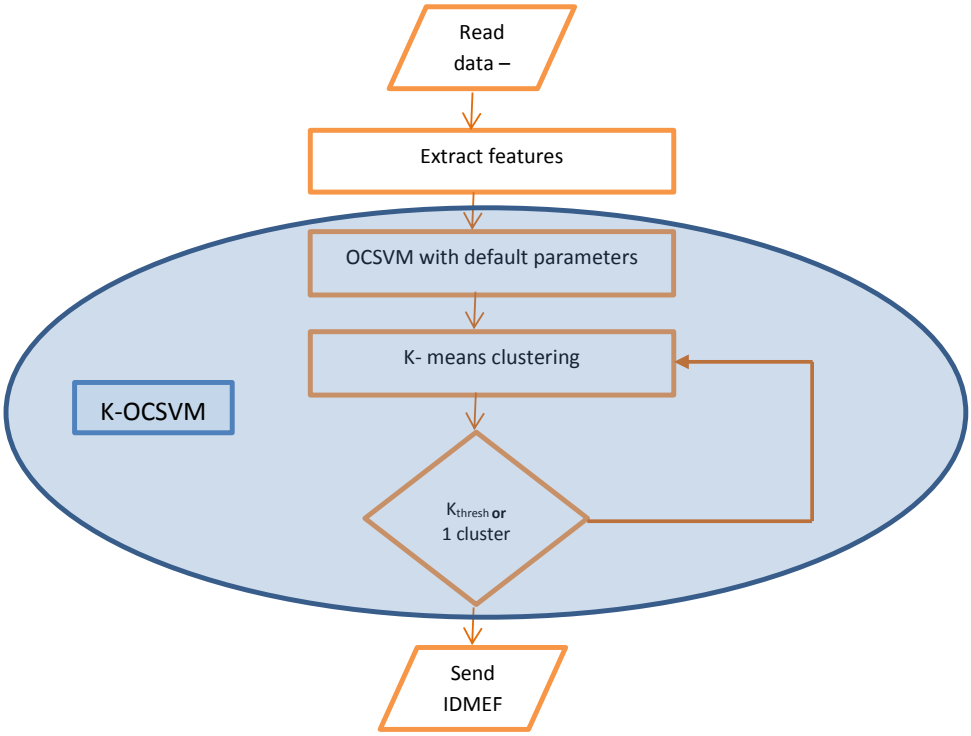

Fig. 3.1 K-OCSVM module

The OCSVM classifier runs with default parameters and the outcome consists of all possible outliers. These outliers are clustered using the k-means clustering method with *2* clusters, where the initial means of the clusters are the maximum and the minimum negative values returned by the OCSVM module. From the two clusters that are created from the K-means clustering, the one that is closer to the maximum negative value (severe alerts) is used as input in the next call of the K-means clustering. This procedure is repeated until all outcomes are put in the same cluster or the divided set is big enough when compared to the initial one, according to the threshold parameter k_{thres}.

The K-means clustering method divides the outcomes according to their values and those outcomes with the most negative values are kept. That way, after the completion of this recursive procedure, only the most severe alerts are communicated from the K-OCSVM. The division of the data need no previous knowledge about the values of the outcomes which may vary from *-0.1* to *-160* depending on the values assigned to the parameters σ and ν. The method can find the most important/possible outliers for any given values of parameters σ and ν.

As it shown in a later Chapter, the K-OCSVM method performs well in terms of low overhead and has good accuracy. K-OCSVM can detect all important variations in network traffic that may constitute an attack, but it achieves this by ignoring small fluctuations that may obscure low probing attacks. For this reason, and based on the high requirements of the CockPitCi project, a novel method that achieved both high accuracy and low overhead was developed. This new method, namely IT-OCSVM, is described in the next section.

3.3 IT-OCSVM Idea

Behavioral analysis is the

Social network analysis (*SNA*) can be used in order to discover security policy breaches in a network and refers to the use of network theory to analyze social networks. That is, it views social relationships in terms of network theory, consisting of nodes, representing individual actors within the network, and ties which represent relationships between the individuals, such as friendships, kinships, organizations and sexual relationships. By using comparative metrics of a network's structure during normal and abnormal operation, we can discover security policy breaches. One can assume that network communication between nodes, constitutes a social network of users and their applications, so the appropriate methods of social network formal analysis can be applied [91]. In on-line social systems perpetrators of malicious behavior often display patterns of interaction that are quite different from regular users, which can be identified through the application of anomaly detection techniques. Thus, in accordance [92], network anomalies can be defined as patterns of interaction that significantly differ from the norm and in order to capture the appropriate patterns of interaction, specific aspects of entities' behavior are used (e.g. email analysis).

Deviation from the normal protocol operation in communication networks has received considerable attention from the research community in recent years. A malicious node may use the vulnerabilities of the system architecture to perform different kinds of attacks. Hence, considering network reliability it is necessary to develop efficient techniques to detect misbehaving clients in a timely manner and the correlation coefficients between entities can

effectively detect malicious nodes [93]. Similar to this, in a SCADA system, individual entities demonstrate a quite different communication behavior when infected, in terms of mean packet generation frequency (traffic burst), protocol distribution or interaction pattern.

Discovering anomalies in the context of a network system is a challenging issue due to the complexity of the environment and the different nature of the induced attacks. Regarding node behavior related decisions it makes sense to ask more than one decision mechanism, since this practice assures a more trusted final decision. Ensemble systems of classifiers are widely used for intrusion detection in networks [94]. These aim to include mutually complementary individual classifiers, which are characterized by high diversity in terms of classifier structure [85], internal parameters [86] or classifier inputs[87].

As stated in the previous chapter one of the main objectives when implementing an IDS, along with fast response and high accuracy is low overhead. Since the solution of adding a control channel that is dedicated to the messages that are produced from the IDS, an aggregation mechanism that groups initial alerts and sends a limited number of messages reporting the fault/intrusion accurately and on time was also developed.

The present chapter presents a recently proposed intrusion detection mechanism, namely the IT-OCSVM [95], against the baseline OCSVM method. The mechanism uses a central OCSVM and a cluster of automatically produced ones, one for each source that induces signifficant traffic in the system, an embedded ensemble mechanism, social metrics, an aggregation method and a K-means clustering procedure that categorizes aggregated alerts. The mechanism runs in a distributed way and produces dedicated IDMEF (Intrusion Detection Message Exchange Format) messages that report the severity of the attack detected. The proposed mechanism is incorporated in a distributed IDS communicating with other detection and management components of the system.

3.4 IT-OCSVM Implementation

The main purpose of the *IT-OCSVM* detection mechanism is to perform anomaly detection in a time-efficient way, with good accuracy and low overhead, within a temporal window adequate for the nature of SCADA systems. In order to achieve the aforementioned goals several operation stages need to be carried out (See Figure 3.2).

- **Pre-processing** of raw input data in order to feed the *IT-OCSVM* module. The attributes in the raw data of the testbed contain all forms: continuous, discrete, and symbolic, with significantly varying resolution and ranges. Most pattern classification methods are not able to process data in such a format. Hence, pre-processing is required before pattern classification models can be built. This consists of two steps: the first involves

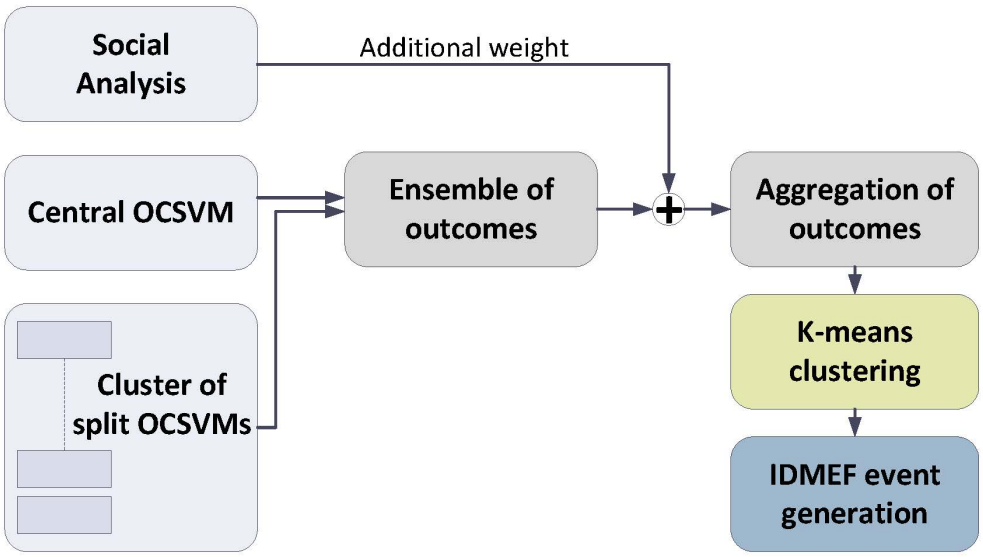

Fig. 3.2 Architecture of detection mechanism

mapping symbolic-valued attributes to numeric-valued attributes and the second is implementing scaling. Different pre-processing techniques are used based on the characteristics of each feature type [96].

- **Selection of the most appropriate features** for the training of the *IT-OCSVM* model. These features are divided into content and time based features. Since the majority of DoS and probing attacks may use hundreds of packets, time-based features are mostly used.

- **Creation of cluster of OCSVM models** that are trained on discrete sources. There are many slow probing attacks that scan the hosts using much larger intervals, thus being able to merge into the overall traffic in the network. As a consequence, these attacks cannot be detected using derived time-based features and in order to capture them, the raw data after arriving in the module is split into different datasets according to the sender of the packet. An OCSVM module is created and trained for each split dataset. The cluster of split OCSVMs run in parallel with the central OCSVM and produce errors targeted to the specific source.

  It is important to mention that a split OCSVM is not created for each source, but only for those sources that produce high traffic in the network during the training period. In order to separate significant nodes a threshold $P_{packets}$ is used and every source that produces a number of packets over this threshold during the training period is marked

as a significant node. If during the testing period a source is showing big activity, while not being marked as significant, a medium alarm is fired for it.

- **Testing of the traffic dataset that contains malicious attacks**. Based on the models created from the training phase the new dataset is tested against normal patterns. Each OCSVM module returns a function f that takes the value +1 in a region capturing normal data points (i.e. for events implying normal behaviour of the SCADA system) and takes a negative value elsewhere (i.e. for events implying abnormal behaviour of the SCADA system).

- **Ensemble of classifiers** The initial outcomes of the different OCSVM modules are combined by the ensemble based mechanism that uses mean majority voting.

- **Social analysis** Social network analysis is executed based on the network traces and Spearman's rank correlation coefficient is used in order to add weight to alerts produced from different sources.

- **Fusion of the information** Due to the possible existence of multiple anomalies in a SCADA system, the outcomes of the different models are gathered and classified in terms of importance. This importance is derived through aggregation and k-means clustering of the outputs.

- **Communication of the mechanism**. In order to cooperate with other components of the IDS the mechanism sends IDMEF files. The created files describe the nature of the alert, in terms of importance, the position in the system, time etc.

A more detailed description of the proposed method and its evaluation after the deployment in realistic SCADA systems can be found in [97, 95, 23]. In the next subsection the social analysis module that is embeded in the IT-OCSVM method is described.

3.5 Summary

This chapter presents the two novel methods, namely the K-OCSVM and the IT-OCSVM. The detection mechanisms, which run in a distributed way, can be used in large SCADA networks with no additional modifications.

The main feature of the K-OCSVM module is that it can perform anomaly detection in a time-efficient way, with good accuracy and low overhead. Low overhead is an important evaluation metric of a distributed detection module that is scattered in a real-time system, since frequent communication of IDMEF files from detection agents degrade the performance

of the SCADA network. On the other hand, since sometimes these fluctuations obscure real attacks, better methods that can classify them, instead of ignoring them was needed. This need led to the development of the second method, namely the IT-OCSVM.

Speaking for the IT-OCSVM, the combination of social network analysis metrics with machine learning classification techniques, enhances the performance of the detection mechanism and improves accuracy for all the simulation scenarios investigated. Moreover, the aggregation procedure embedded in the proposed mechanism decreases the overhead of the IT-OCSVM and makes it easily incorporable in a soft real time system. That is, the method produces a small amount of final alerts and manages to detect all the simulated attacks.

Chapter 4

Evaluation

In this chapter we present the different testbeds that were used in order to evaluate the performance of the proposed methods. Also all the experiments that were conducted, offline and online, and the several performance metrics are presented and critically analyzed.

4.1 Testbeds and Experiments

4.1.1 Small-scale SCADA System

The testbed architecture mimics a small-scale SCADA system, comprising the operations and field networks, which include a Human-Machine Interface Station (for process monitoring), a managed switch (with port monitoring capabilities for network traffic capture), and two Programmable Logic Controller Units, for process control.

Three kinds of attacks are evaluated:

- **Network scan attack** In a typical network scan attack, the attacker uses TCP/FIN scans to determine if ports are closed to the target machine; closed ports answer with RST packets while open ports discard the FIN message. FIN packets blend with background noise on a link and are hard to detect.
- **ARP cache spoofing - MITM attack ARP cache spoofing** is a technique where an attacker sends fake ARP messages. The aim is to associate the attacker's MAC address with the IP address of another host, thus causing any traffic meant for that IP to be sent to the attacker instead. The attacker could choose to inspect the packets, modify data before forwarding (**man-in-the-middle attack**) or launch a denial of service attack by causing some of the packets to be dropped.

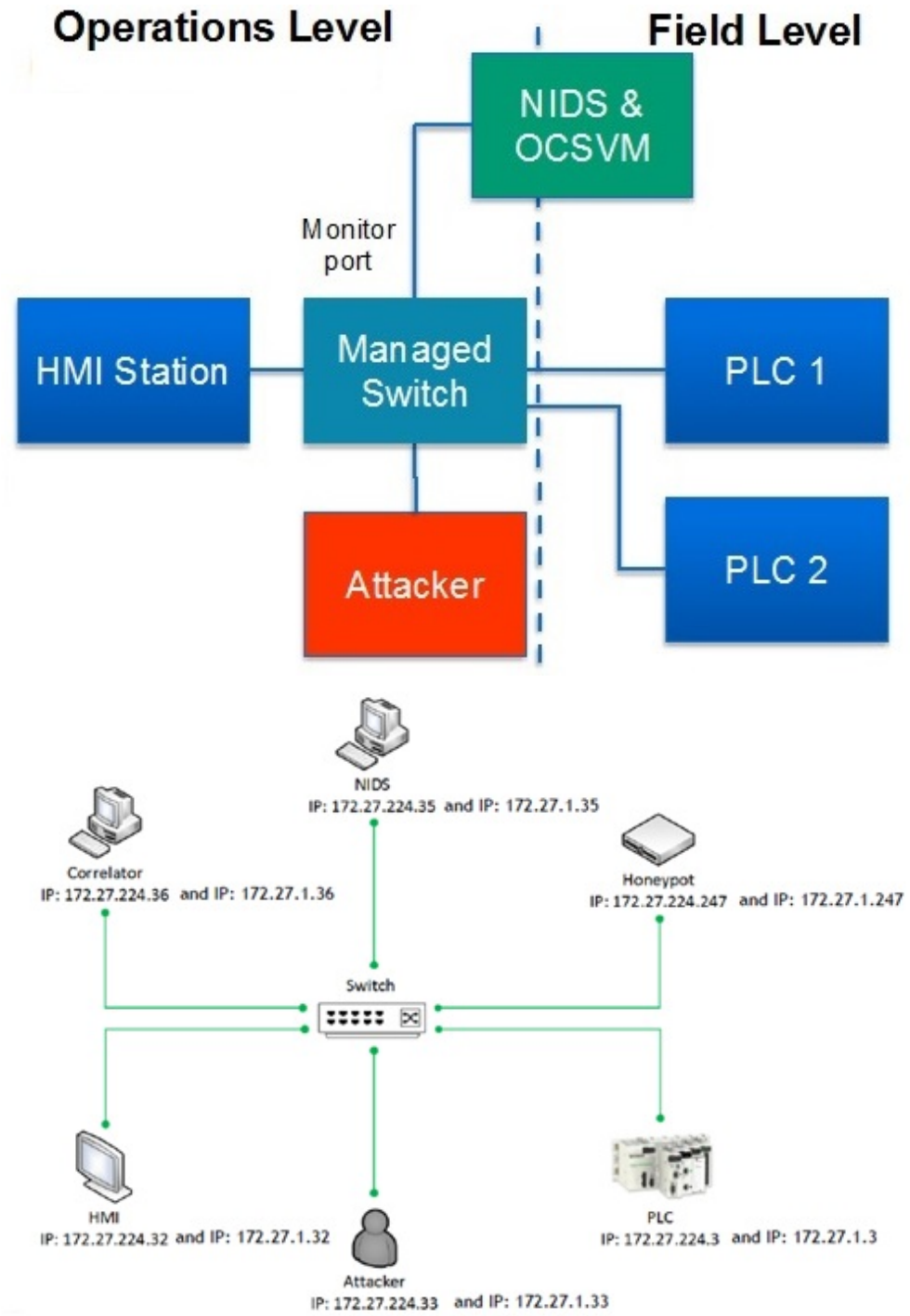

Fig. 4.1 Architecture of the testbed

- **DoS attack** Network flood refers to the situation where the attacker floods the connection with the PLC by sending SYN packets.

4.1.2 Hybrid Test Bed (HTB)

The Hybrid Environment for Development and Validation (HEDVa) was designed by the Israeli Electric Company for the development and validation of CIs as well as security research projects [23], being used to implement the attack use cases for the PIDS validation effort. Its architecture (see Figure 4.2) constitutes a hybrid approach, because of the existence of real/physical SCADA and network/telecom infrastructure components, which are used to implement a simulation model of the electric grid elements.

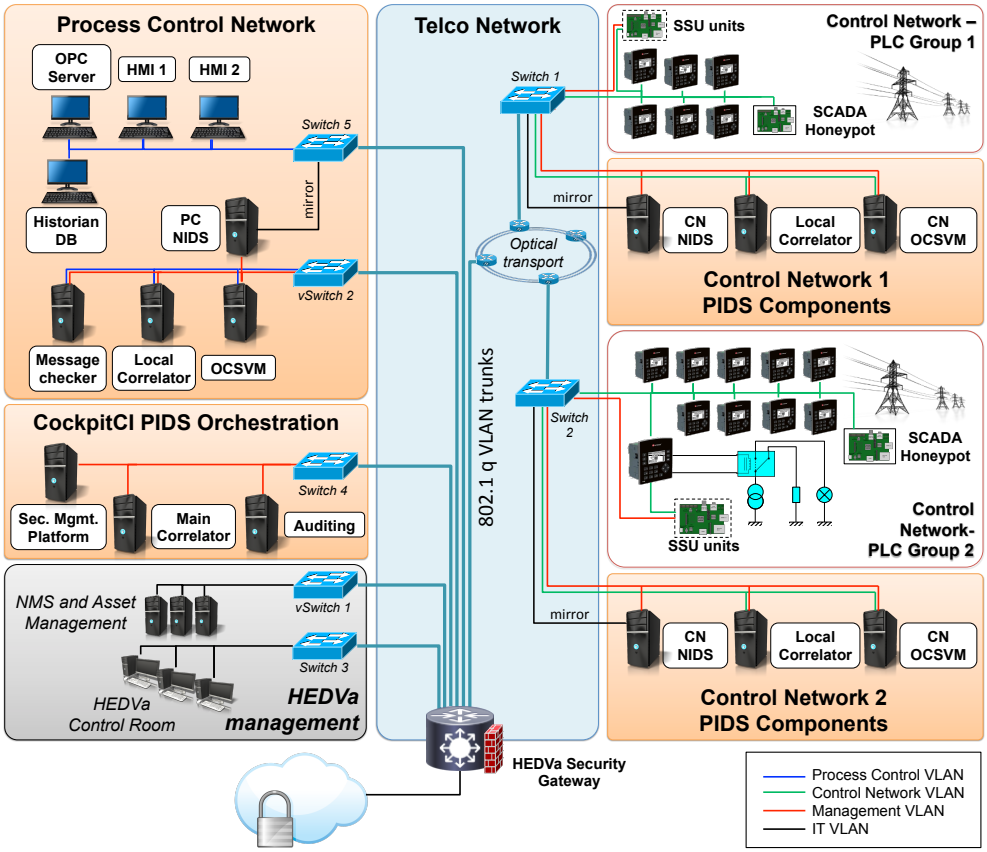

Fig. 4.2 HEDVa networking architecture

The CockpitCI emulated HEDVa scenario corresponds to an energy distribution grid, as shown in Figure 4.3. Its implementation required the development and testing of different modeling techniques, also including related key performance indicators, fine-tuned with data from production scenarios. Grid elements such as breakers and feeders are controlled by real PLCs in a simulation model, with voltage and current values for segments being calculated accordingly using a mathematical model of the physical grid. In case of failure, the scenario is reconfigured accordingly with the operator's Fault Isolation and Service Restoration procedures.

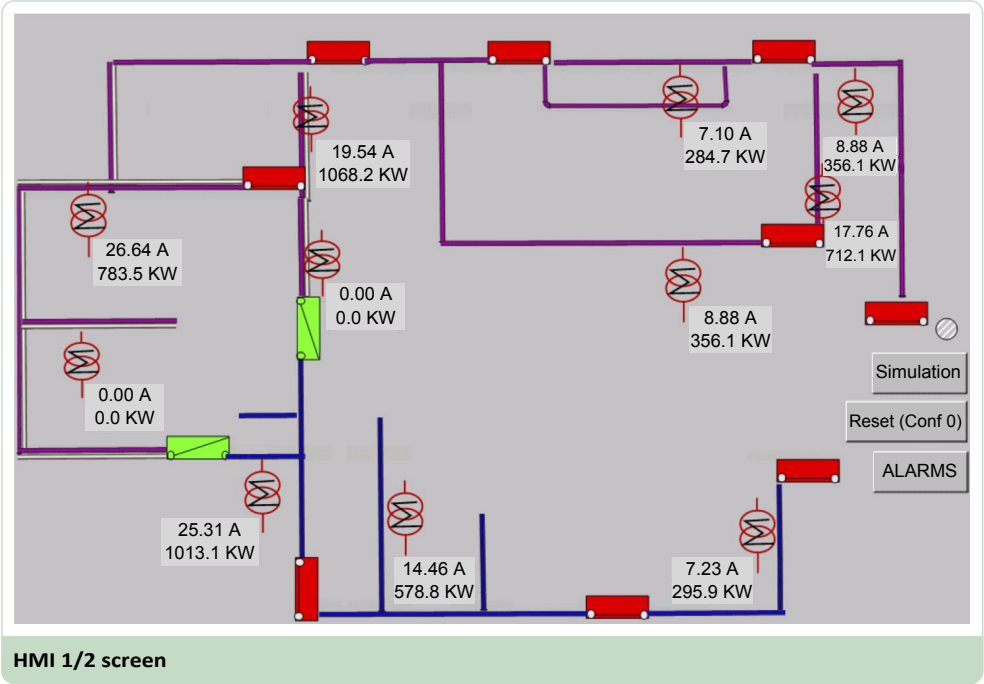

Fig. 4.3 HEDVa grid scenario, with breakers and substation feeders

From an implementation standpoint, the HEDVa scenario makes use of real SCADA and NMS elements, physical components of electrical and telecom infrastructure and emulated elements of electrical and telecom infrastructure, providing a test ground for the CockpitCI PIDS (see Figure 4.2). The Hybrid testbed architecture mimics a medium-scale SCADA system, comprising the operations and field networks and including Human-Machine Interface Stations (for process monitoring), six managed switches (with port monitoring capabilities, for network traffic capture), and several Programmable Logic Controller Units, for process control. The initial dataset consists of over *3* million rows, each representing a packet sent in the system, capturing network of several days. The dataset is split in *65* smaller ones of *50.000* rows. The datasets that were used contain several attacks.

4.2 K-OCSVM Method Evaluation

We evaluate the performance of the method using data from a testbed that mimics a small-scale SCADA system and from a Hybrid testbed of a medium sized SCADA system. The parameters used for the evaluation of the performance of K-OCSVM are listed in Table 4.1.

Parameter	*Range of Values*	*Default value*
σ	0.1 - 0.0001	0.007
ν	0.002 - 0.05	0.01
k_{thresh}	2 - 3	2

Table 4.1 Evaluation Parameters

4.2.1 Testbed Scenario

The second trial is conducted off line with the use of two datasets extracted from the testbed that we described earlier.

In Table 4.2 we show the number of alert messages (IDMEF) sent from OCSVM and K-OCSVM respectively. From this table it is shown how parameters σ, ν affect the performance of OCSVM for the testbed scenario. While for the same network trace file OCSVM produces from *10529* to *10704* alert messages according to the values of the parameters, K-OCSVM produces the same *120* alert messages. All the reported attacks are concerning the *DoS* attack that creates the biggest fluctuation in the network traffic.

Parameter σ	*Parameter* ν	*K-OCSVM*	*OCSVM*
0.007	0.002	120	10529
0.007	0.01	120	10703
0.007	0.005	120	10584
0.0001	0.01	120	10602
0.1	0.01	120	10704

Table 4.2 Performance evaluation of K-OCSVM and OCSVM for $K_{thres} = 2$

4.2.2 Hybrid Testbed Scenario

The third trial is conducted off line with the use of large datasets extracted from a Hybrid Testbed (HTB) scenario. Both OCSVM and K-OCSVM are trained and tested with these datasets, using cross validation. The mean number of alert messages sent by the two modules is shown in Table 4.3. The datasets that were used contain several attacks, but the ones presented in this subsection contain only data from a normal operation of the HTB

Using real datasets of a medium sized HTB SCADA system the performance of the proposed K-OCSVM method is very stable compared to a simple OCSVM under the same configuration. This behavior is very promising since K-OCSVM method has a very low false

Parameter σ	*Parameter* ν	*K-OCSVM*	*OCSVM*
0.007	0.002	1 - 2	40
0.007	0.01	1 - 2	207
0.007	0.005	1 - 2	105
0.0001	0.01	1 - 2	85
0.1	0.01	1 - 2	271

Table 4.3 Performance evaluation of K-OCSVM and OCSVM for $K_{thres} = 2$

alarm rate (*lower than 0.02 %*) while at the same time the overhead induced by the method is negligible.

Computational Cost and Time Overhead

Complexity of an intrusion detection system can be attributed to hardware, software and operation factors. For simplicity, it is usually estimated as the computing time required to perform classification of the dataset and output the final alarms. In order to evaluate the complexity of the proposed method, we calculate the execution time and compare it to a simple OCSVM module. The evaluation is conducted on a PC with Intel core *2* duo *1.7* Mhz CPU, *2*GB main memory, *80*GB hard disk *7200* rpm hard disk and Microsoft Windows *7 64*bit. In Figure 4.4 we represent the time performance of the method compared to a simple OCSVM module for the testbed scenario.

According to Figure 4.4 execution time of the proposed K-OCSVM is slightly bigger compared to a simple OCSVM method. The performance gap is around *5%* to *10%* for all the datasets used in the simulation. Based on these observations we conclude that the system, performs a classiffication in a comparable time to that of a simple OCSVM classifier, and it thus can be adopted in soft real-time applications. We have to mention that the performance evaluation which is conducted in this subsection, does not include the time that each detection mechanism needs in order to create and disseminate IDMEF messages. It is evident that the OCSVM classifer, compared to the proposed K-OCSVM, needs significant additional time in order to send all the detected alarms.

4.3 IT-OCSVM Method Evaluation

We evaluate the performance of the method using data from a testbed that mimics a small-scale SCADA system and from a Hybrid testbed of a medium sized SCADA system.

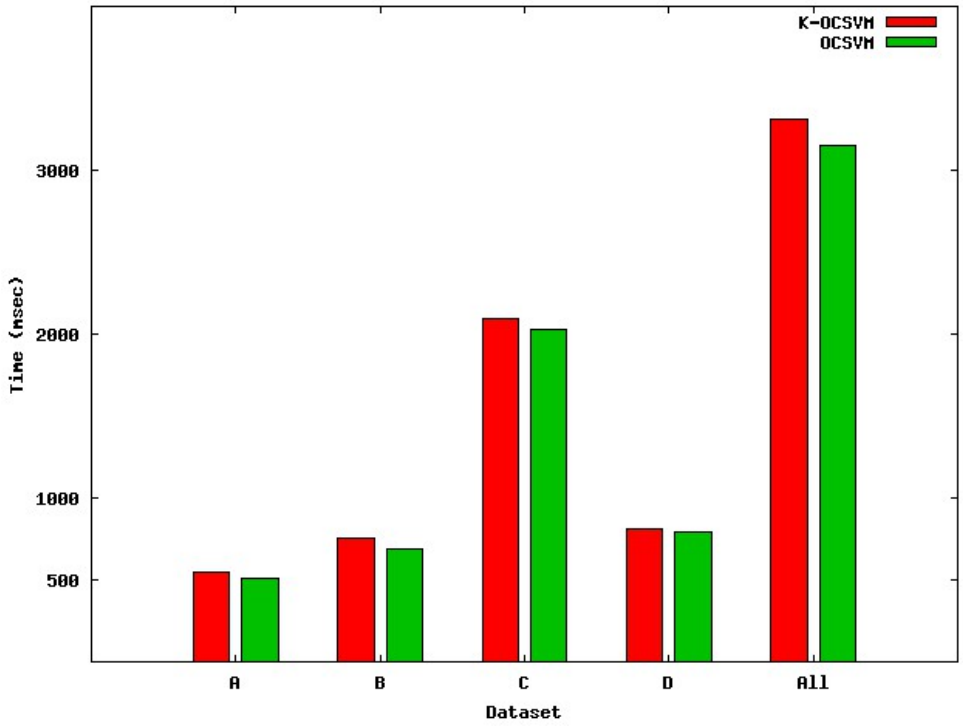

Fig. 4.4 Computational cost for the testbed scenario

4.3.1 Testbed Scenario

IT-OCSVM is evaluated using datasets extracted from a testbed that mimics a small scale SCADA system under normal and malicious operation. The evaluation of the proposed method attests the superiority of the new structure in terms of accuracy, false alarm rate and system overhead. The trial is conducted off line with the use of two datasets extracted from the testbed (Figure 4.1). The parameters used during evaluation of the proposed IT-OCSVM detection method are shown in table 4.4.

Table 4.4 Evaluation parameters

IndependentParamater	*Default value*
ν	0.001
σ	0.01
$P_{packets} - (No\ of\ split\ OCSVMs)$	$\frac{1}{100} - (5)$
Ensemble mechanism	*Weighted sum value*
$k-means$	2 *stages*

This subsection describes the performance of the proposed OCSVM based intrusion detection algorithm for the simulated data. During the testing period several attack scenarios are simulated in the testbed, which include: a network scan, a network flood and an MITM attack. Since the attacks are performed during different time periods we divide the testing dataset into several smaller ones, each containing a different attack. The testing data consists of normal and attack data and the composition of the data sets is as follows:

- Testing set-A' : 1 - 5000: Normal data
- Testing set-B' : 5000 - 10000: Normal data + **Arp spoofing** attack + **Network scan**
- Testing set-C' : 10000 - 25000: Normal data + **Flooding DoS attack** + **Network scan**
- Testing set-D' : 25000 - 41000: Normal data + **MITM attack**

Layer 2/3/4 Attacks

These scenarios correspond to malicious activities targeting the SCADA network resources associated to the data-link (2), network (3) and transport layers (4) of the OSI model. For this category, two validation use cases were chosen: a FIN/SYN Scan and a SYN Flooding.

A **port scan** is a typical first step for an attacker to obtain information about the network's hosts and its topology, using techniques such as SYN, ACK or FIN scans (among others). This attack was validated with the use of SYN and FIN probes (see Figure 4.5) to devices on a network scope, to scan for open TCP/IP ports. The latter also being known as "stealth scans", as some firewalls may log them SYN ("half-open") attempts to restricted ports. Moreover, a FIN packet sent to a closed port on certain hosts (mostly UNIX-based platforms and most PLC/devices) will pass undetected, generating a RST response or being ignored for an open port. Figure 4.5 illustrates such a scenario, where an attacker sends a series of FIN or SYN probes (green flows) to devices on a network scope and waits for the responses.

The **SYN flooding** Denial-of-Service (DoS) attack abuses TCP connections, by sending a high volume of SYN packets to the PLC/RTU. This creates a large number of half-open connections on the target device, leading to resource exhaustion and the inability to answer new requests.

SCADA Protocol and Process-level Attacks

SCADA protocols can be specifically targeted by corruption, interception or tampering attacks. These can potentially lead to a loss of confidentiality, visibility, or device connectivity, also providing support to implement process-aware attacks, thus compromising safety and

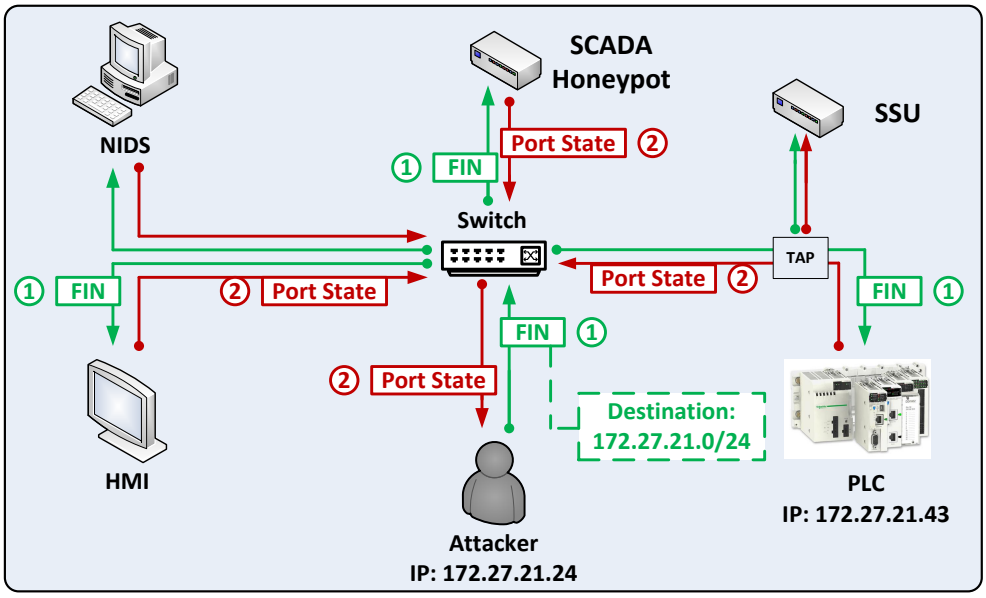

Fig. 4.5 TCP FIN scan

security. A **Man-in-the-Middle (MITM)** attack using Address Resolution Protocol (ARP) poisoning was chosen as the validation use case for this category, exercising protocol and process-level detection capabilities. A MITM allows for a third-party to become involved in the middle of a communication stream, while remaining unnoticed, eventually taking advantage of process-level knowledge to manipulate SCADA parameters. The MITM use case starts with an intruder host within the Control Network sending unrequested ARP replies to poison the targets' ARP caches and hijack the Modbus TCP protocol flows between the HMI 1 system and the PLCs; this allows for the attacker to eavesdrop communications and gather information about the SCADA network and the process. Afterwards, the attacker will use the previously gathered information to craft a scripted real-time simulator using the *Scapy* tool (see Figure 4.6 - green flows), reproducing the normal operation of the simulated grid and providing nominal status data back to the HMI 1, while directly manipulating the PLCs, without any visible supervisory feedback. MITM attacks can leverage several techniques from ARP poisoning to routing redirection.

The ARP-poisoning based MITM attack was chosen as the validation use case for this attack category, as it exercises both protocol and process-level cyber detection capabilities. In the ARP MITM, an attacker generates unrequested ARP replies for both the HMI and the PLC, poisoning the local ARP caches in such a way that the attacker host places itself in the middle of the communications path. For the PLC, the MAC address of the attacker system becomes associated with the IP of the HMI, and for the HMI, the MAC of attacker associates with the IP of the PLC. In this way, the attacker places itself in the middle of the

communications path. In the first stage of an ARP poisoning MITM, the attacker generates a series of unrequested ARP replies for both the HMI and the PLC (top half), poisoning the local ARP caches in such a way that the MAC address of the attacker system becomes associated with the IP of the HMI for the PLC and the IP of the PLC for the HMI, respectively. Further interaction attempts from the HMI to the PLC will be redirected to the attacker system, and vice-versa.

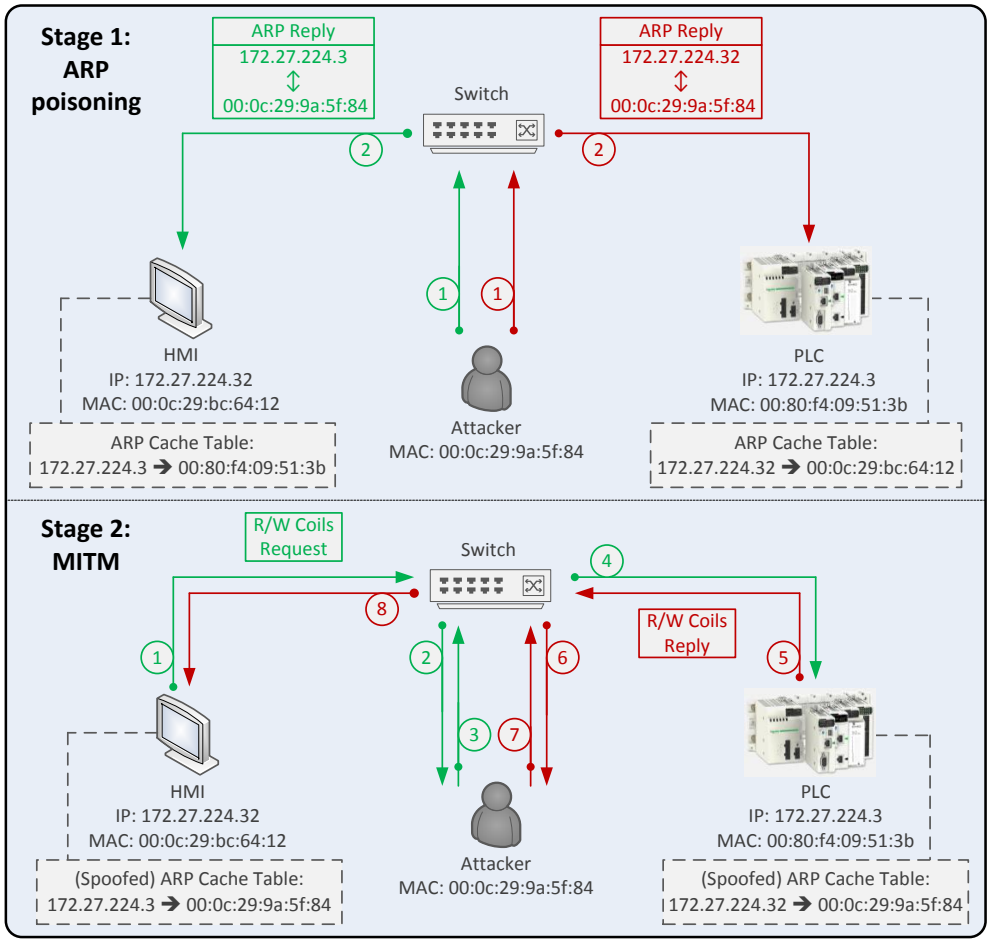

Fig. 4.6 ARP-based, Man-in-the-Middle attack

The IT-OCSVM, performs well in terms of detection accuracy and false positive rate as shown in Table 4.5.

Impact of the Fusion Mechanism

The fusion of the alarms produced by the individual OCSVMs consists of two stages. The first is aggregation of alarms per source and the second is the clustering of them using a two stage K-means clustering algorithm. The outcome of the fusion procedure is the deduction of

Table 4.5 Performance evaluation of the IT-OCSVM module

Dataset	*DA*	*FAR*
A	98.81%	1.18%
B	94.6%	3.25%
C	95.20%	1.51%
D	96.37%	2.3%
All	96.3%	2.5%

the communicated alarms in the system and also their classification as: possible, medium and severe. In Table 4.6 the number of initial and final aggregated alarms is presented. It can be observed that the number of final alarms is significantly lower compared to the initial ones, thus reducing the communication costs that such a distributed mechanism has in the network.

Table 4.6 Aggregated alarms produced by IT-OCSVM are significantly decreased compared to the initial ones

Dataset	*Initial alarms*	*Aggregated alarms*
A	129	16
B	658	21
C	9273	18
D	203	16
All	10507	22

The IT-OCSVM system categorizes alarms according to the level of severity they have, with most being classified as possible and those few originating from real attacks in the system are termed severe (see Figure 4.7). Since the proposed mechanism is part of a distributed PIDS, the information sent by the IT-OCSVM can be combined with those sent by the other detection modules. For this reason this categorization of the alarms is important, if correct final decisions about the situation in the network are to be taken.

Computational Cost and Time Overhead

The complexity of an IDS can be attributed to hardware, software and operational factors. For simplicity, it is usually estimated as the computing time required to perform classification of the dataset and output the final alarms. Increasing the number of classifiers usually increases the computational cost and decreases their comprehensibility. For this reason, special care

must be taken when choosing parameter $P_{packets}$. This parameter determines the number of created split datasets and thus, split OCSVMs. While the increase in number of split OCSVMs does not impose any significant increase in the method's performance, this may slow down the detection mechanism.

In figure 4.8, we illustrate the time performance of the method compared to a simple OCSVM. The evaluation was conducted on a PC with Intel core 2 duo 1.7 Mhz CPU, 2GB main memory, 80GB hard disk 7200 rpm hard disk and Microsoft Windows 7 64bit.

According to Figure 4.8, the execution time of the proposed IT-OCSVM is bigger compared to a simple OCSVM method. However, for the extreme configuration where *15* OCSVMs are created the performance gap increases and the proposed IT-OCSVM is *75 %* slower than the simple OCSVM, whereas when the IT-OCSVM operates under the default configuration (*5-6* OCSVMs) the performance gap is *55 %*. Based on these observations we conclude that the system, under the default configuration, performs a classification in a comparable time to that of a simple OCSVM classifier and thus, it can be adopted in soft real-time applications.

We have to mention that the performance evaluation conducted in this subsection, does not include the time that each detection mechanism needs in order to create and disseminate IDMEF messages or the computational and time costs that the correlator needs in order to collect and analyze the alarms. It is evident that the OCSVM classifier, compared to the proposed IT-OCSVM, needs significant additional time in order to send all the detected alarms. Moreover, the categorization that is performed by the IT-OCSVM mechanism reduces the computational load of the correlator that collects the alarms from the distributed detection agents.

4.3.2 Hybrid Testbed Scenario

After the PIDS system was successfully installed, verified and validated the next step was the evaluation of the validation results and recommendations for the next steps. During the CockpitCI project evaluation, the conclusions of the different scenarios were provided, and the results of the cyber-attack scenarios were analysed. The analysis included the results of scenarios implementated for all the levels of ICS described in this document as also, was all the equipment used in the CockpitCI Validation system.

Use Case: Network FIN SCAN

Similar to the procedure described earlier for the small testbed scenario, a FIN scan attack was initiated using the nmap tool. This attack produced a network trace, a snapshot of which

is shown in Figure 4.9. The OCSVM managed to identify this attack and create alarms, creating IDMEF files, as shown in Figure 4.10.

Use Case: Network SYN Flood

The SYN attack, illustrated in Figure 4.11 explores the nature of TCP connections, by sending a series of SYN packets (signaling the intention to initiate a TCP connection) to the PLC/RTU, to which it responds with a SYN-ACK that is never acknowledged (ACK) by the sender.

The captured network flows for the attacks were reported from the OCSVM module with the use of IDMEF flies.

Use Case: ARP Cache Overwrite Attack

Similar to the procedure described in the testbed scenario earlier, and ARP spoofing attack the attacker manages to associate his MAC address with the IP of the HMI for the PLC and the IP of the PLC for the HMI, respectively. This means that further interaction attempts from the HMI to the PLC will be redirected to the attacker system, and vice-versa. Working with network traces that are produced during the ARP spoofing attack the OCSVM module managed to detect this attack and communicate to the PIDS through IDMEF files.

Other attacks that were conducted during the validation of the PIDS, such as honeypot interaction, infrastructure attack and RAT installation, were detected from other detection mechanisms like SNORT, shadow RTU, Software Checker component, etc. Due to the high volume of packets that were captured during the validation of the PIDS, which were unlabeled, metrics like accuracy and false positive rate were not calculated and only the capability of the PIDS to report the attacks real time was validated.

4.4 Summary

In this chapter we presented firstly the evaluation of an intrusion detection module for SCADA systems that is based on an OCSVM classifier and a recursive K-means clustering method. The module is trained off-line by network traces, after the attributes are extracted from the network dataset. The K-OCSVM method is based on a recursive K-means clustering and reassures that small fluctuations on network traffic, which most of the times cause OCSVM to trigger false alarms, are ignored by the proposed detection module. The added computational time of the method compared to a simple OCSVM varies between 5% and 10%, which results in a negligible time overhead. This overhead does not include the time that each detection

mechanism needs in order to create and disseminate IDMEF messages. By adding the time needed in order to create and send each IDMEF file to the IDS management system the proposed K-OCSVM method prevails on the overall performance in terms of time efficiency. Finally we investigated how the parameter k_{thres} affects the performance of the method and proposed a multi-stage K-OCSVM method for better accuracy.

This chapter also analyzed the IT-OCSVM mechanism and evaluated it against the baseline method for different attack scenarios. The detection mechanism, which runs in a distributed way, can be used in large SCADA networks with no additional modifications. The combination of behavioral analysis with machine learning classification techniques, enhances the performance of the detection mechanism and improves accuracy for all the simulation scenarios investigated. Moreover, the aggregation procedure embedded in the proposed mechanism decreases the overhead of the IT-OCSVM and makes it easily incorporable in a soft real time system. That is, the method produces a small amount of final alerts and manages to detect all the simulated attacks. The intrusion detection module is part of a distributed IDS system.

The presented methods can be used in soft real time situations as components of a PIDS, having good performance in terms of time overhead and accuracy. Especially IT-OCSVM shows high accuracy for all the network based attacks that were launched both in the small SCADA testbed and the in the HEDVa emulator. Other kind of attacks, like honeypot interaction, infrastructure attack and RAT installation cannot be detected from the proposed methods since they demand deep packet inspection.

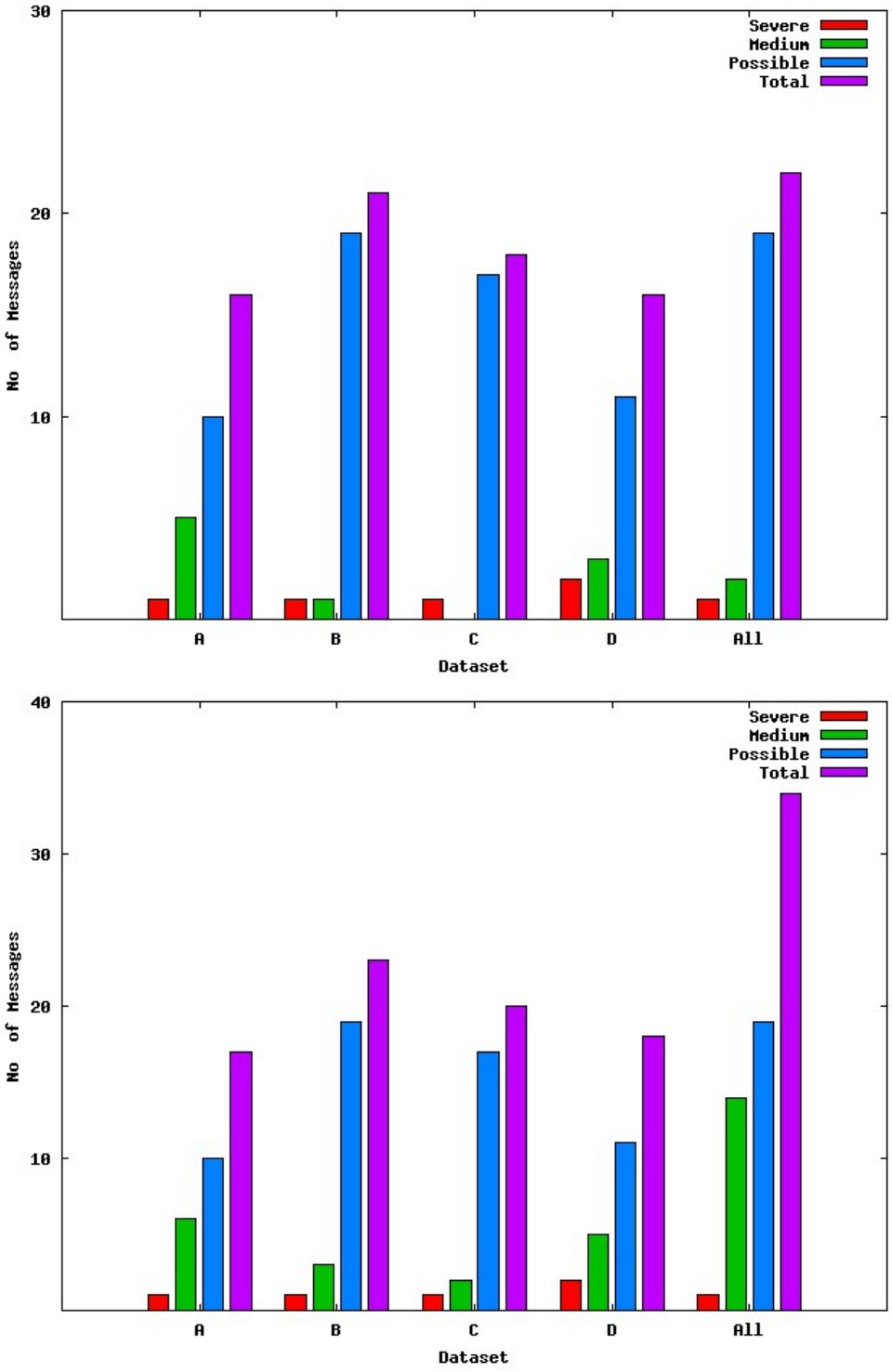

Fig. 4.7 IT-OCSVM categorizes aggregated alarms. The left diagram shows aggregated alarms created by IT-OCSVM without the additional medium alarms and the right diagram illustrates all the final alarms created by the IT-OCSVM

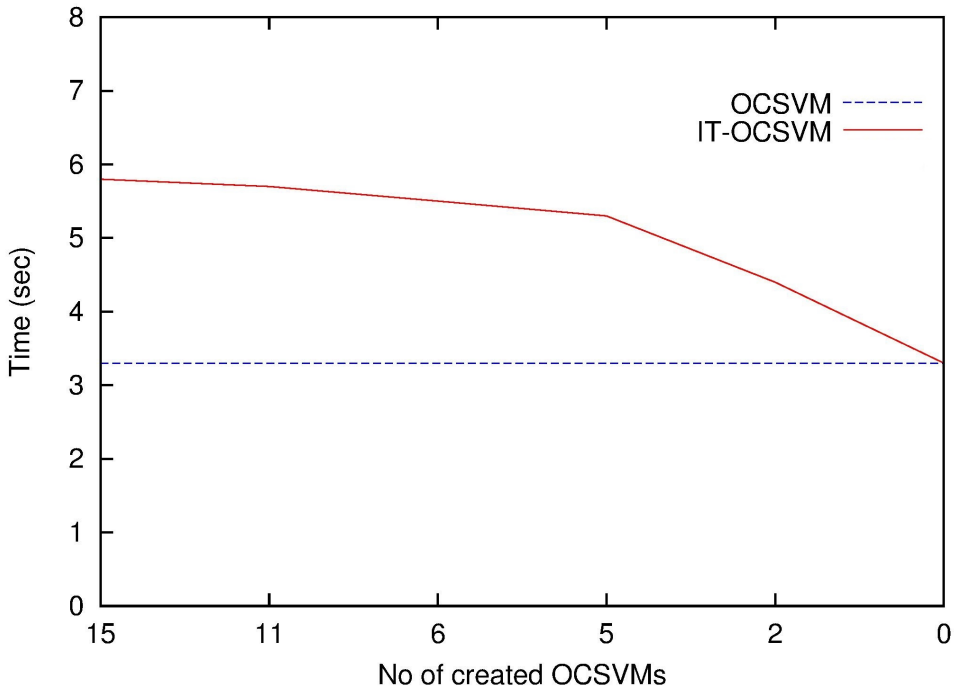

Fig. 4.8 Approximate execution time for the entire testing dataset

```
[root@nids]#tcpdump -nnNeti eth0 'tcp[tcpflags] & tcp-fin !=0'
17:39:40.754402 IP 172.27.21.24.35310 > 172.27.21.41.http: Flags [F],
seq 3038984269, win 4096, length 0
17:39:40.754410 IP 172.27.21.24.35310 > 172.27.21.41.ssh: Flags [F],
seq 3038984269, win 2048, length 0
17:39:40.754415 IP 172.27.21.24.35310 > 172.27.21.41.newacct: Flags
[F], seq 3038984269, win 4096, length 0
17:39:40.754422 IP 172.27.21.24.35310 > 172.27.21.41.asa-appl-proto:
Flags [F], seq 3038984269, win 4096, length 0
17:39:40.754429 IP 172.27.21.24.35310 > 172.27.21.41.hostname: Flags
[F], seq 3038984269, win 4096, length 0
17:39:41.805577 IP 172.27.21.24.35311 > 172.27.21.41.hostname: Flags
[F], seq 3038918732, win 3072, length 0
17:39:41.805589 IP 172.27.21.24.35311 > 172.27.21.41.asa-appl-proto:
Flags [F], seq 3038918732, win 3072, length 0
17:39:41.805593 IP 172.27.21.24.35311 > 172.27.21.41.newacct: Flags
[F], seq 3038918732, win 1024, length 0
17:39:41.805598 IP 172.27.21.24.35311 > 172.27.21.41.ssh: Flags [F],
seq 3038918732, win 1024, length 0
17:39:41.805605 IP 172.27.21.24.35311 > 172.27.21.41.http: Flags [F],
seq 3038918732, win 2048, length 0
```

Fig. 4.9 FIN scan network trace

```
<?xml version="1.0" encoding="UTF-8"?>
- <idmef:IDMEF-Message version="1.0" xmlns:idmef="http://iana.org/idmef">
  - <idmef:Alert>
    - <idmef:Analyzer analyzerid="test">
      - <idmef:Node category="unknown">
          <idmef:location>IT Network</idmef:location>
          <idmef:name>OCSVM</idmef:name>
        </idmef:Node>
      </idmef:Analyzer>
      <idmef:CreateTime ntpstamp="0x1130fdd3.0xa0000000">2014-08-19T16:19:43+01:00</idmef:CreateTime>
    - <idmef:Source>
      - <idmef:Node>
        - <idmef:Address category="ipv4-addr">
            <idmef:address>172.27.224.32</idmef:address>
          </idmef:Address>
        </idmef:Node>
      </idmef:Source>
    - <idmef:Target>
      - <idmef:Node>
        - <idmef:Address category="ipv4-addr">
            <idmef:address>172.27.224.3</idmef:address>
          </idmef:Address>
        </idmef:Node>
      </idmef:Target>
      <idmef:Classification text="SEVERE ALARM"/>
    </idmef:Alert>
</idmef:IDMEF-Message>
```

Fig. 4.10 IDMEF message from OCSVM

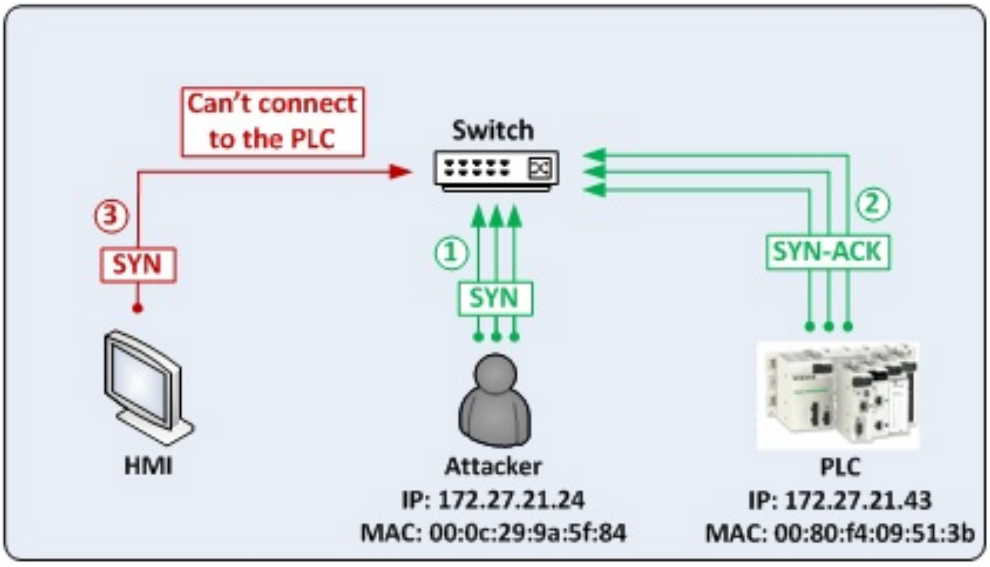

Fig. 4.11 SYN Flood attack

Chapter 5

Conclusions and Future work

5.1 Summary of the Contributions

This commentary addresses the issue of Intrusion Detection in SCADA systems. Two novel IDS that are based on the OCSVM mechanism are presented. Issues such as accuracy, impact of fusion methods, computational and time overhead have been addressed for different datasets that come both from wireless networks and hybrid SCADA test-beds.

The initial method, namely K-OCSVM, can distinguish real from false alarms using the OCSVM method with default values for parameters ν and σ combined with a recursive K-means clustering method. The description of the method along with the evaluation of its performance were presented in one journal article [3] and one conference paper [5]. The K-OCSVM is very different from all previous methods that required pre-selection of parameters with the use of cross-validation or other methods that ensemble outcomes of one class classifiers.The proposed method is tested against the basic OCSVM classifier with different parameters, which attested the stability of the new structure. Due to high requirements that were imposed from the CockpitCi project, both in terms of accuracy and time overhead, a second method, namely IT-OCSVM was developed and validated.

The later method, entitled IT-OCSVM was presented in three journal articles [1, 2, 4]. The IT-OCSVM method is capable of performing outlier detection with high accuracy and low overhead within a temporal window, adequate for the nature of SCADA systems. In order to achieve these goals, the proposed method consists of several distinct stages that need to be carried out, as shown in Chapter 2. The method was tested and validated as a part of the CockpitCI [1] security detection framework for SCADA ICS.

[1] 1CockpitCI FP7 Project (FP7-SEC-2011-1 285647), http://cockpitci.eu

5.1.1 K-OCSVM

We have presented an intrusion detection module for SCADA systems that is based in OCSVM classifier and a recursive k-means clustering method. The module is trained off-line by network traces, after the attributes are extracted from the network dataset. The intrusion detection module is part of a distributed IDS system.

The method is tested on three different datasets. For the first testing scenario, traces of a wireless network are used. This test shows that the method is stable and its performance is not influenced by the selection of parameters ν and σ. For the second scenario, testing of the proposed module is conducted with datasets that are sniffed off a testbed that mimics a small-scale SCADA system under different attack scenarios. After the completion of the test, not only the most important intrusions are detected and reported by K-OCSVM but also the total overhead on the system is limited. Finally, extensive testing of the K-OCSVM module with real datasets, extracted from a medium sized HTB SCADA system, shows that the performance of the proposed K-OCSVM method remains very stable under different configurations. After the execution of the K-OCSVM method, for all the simulated scenarios, only severe alerts are communicated to the system by IDMEF files that contain information about the source, destination, protocol and time of the intrusion.

The main feature of the K-OCSVM module is that it can perform anomaly detection in a time-efficient way, with good accuracy and low overhead. Low overhead is an important evaluation metric of a distributed detection module that is scattered in a real-time system, since frequent communication of IDMEF files from detection agents degrade the performance of the SCADA network. Recursive k-means clustering reassures that small fluctuations on network traffic, which most of the times cause OCSVM to trigger false alarms, are ignored by the proposed detection module. The added computational time of the method, compared to a simple OCSVM, varies between 5% and 10% which results in a negligible time overhead. This overhead does not include the time that each detection mechanism needs in order to create and disseminate IDMEF messages. By adding the time needed in order to create and send each IDMEF file to the IDS management system the proposed K-OCSVM method prevails on the overall performance in terms of time efficiency.

5.1.2 IT-OCSVM

The novel IT-OCSVM method is capable of performing outlier detection with high accuracy and low overhead within a temporal window, adequate for the nature of SCADA systems. In order to achieve these goals, the proposed method consists of several distinct stages that need to be carried out. The first, similar to any other SVM methods, consists of choosing the most

appropriate features for the training of the OCSVM models and the pre-processing of raw input data. In IT-OCSVM alongside the central OCSVM a cluster of split OCSVM models is created. The proposed IT-OCSVM analysis mechanism runs in a distributed way and can be used in any SCADA system after proper training. The performance of the mechanism is enhanced by the combination of behavioral analysis with machine learning techniques and can be easily incorporated into a soft real time detection system, due to its low overhead. Moreover the IT-OCSVM method was tested on a hybrid testbed under attack, as part of a cyber-security framework for SCADA ICS, that was deployed within the CockpitCI project [23].

5.2 Future Work

5.2.1 Future Work, K-OCSVM

The work presented in this commentary can be extended in many ways. For the K-OCSVM method the following extensions can take place.

The proposed K-OCSVM can significantly reduce the produced alarms from the OCSVM module that is the heart of the detection mechanism. The profound advantages of low overhead and low false alarm rate come with the cost of lower accuracy and higher computational overhead. In this section we discuss some enhancements of the proposed method that can improve its performance. The initial results presented in this section can be found in [67].

Parameter k_{thres}

As stated is Chapter 3, K-OCSVM method is a recursive clustering of the alarms produced by the OCSVM module. This recursive procedure is used in order to distinguish severe from possible alarms and finally disseminate only those that represent an actual misbehavior of the system. This way the OCSVM module is enhanced in both the decreased overhead that induces to the system from the disseminated alarm files and in the decreased false alarm rate that it has. The recursive method stops when either all initial alarms are put in the same cluster or when the divided set is big enough compared to the initial one, according to the threshold parameter k_{thres}.

In the simulations presented in Chapter 3 the parameter is set to *2*. When raising the value of this parameter the produced final alarms of the proposed K-OCSVM method raises (See Figure 5.1). This raise also leads to a raise in the false alarm rate. On the other hand the accuracy of the method raises since less profound attacks are detected. By raising the value of the parameter above one limit the method degrades to the initial OCSVM. The optimum

value for parameter K-OCSVM varies according to the architecture of the network. For big, dispersed networks, large values of the parameter would lead to the creation of too many alarms from the module, while at the same time, in a medium sized network, very small values of the parameter would lead to a dangerous decrease of the detection capabilities of the module.

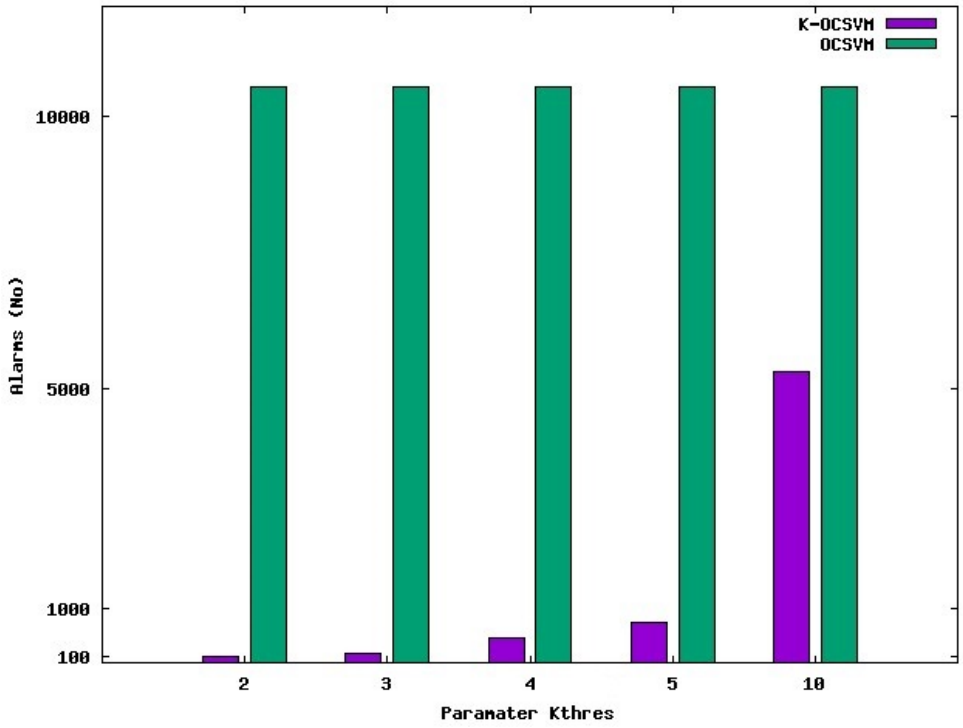

Fig. 5.1 Paramater k_{thres} affects the performance of the K-OCSVM system (Testbed scenario with default parameters σ and ν).

Except from the static configuration of the network, traffic conditions can also affect the performance of the method. In real SCADA systems the network traffic varies between daytime and night, weekdays and weekends. In order to cope with both static and dynamic features of the network an enhanced K-OCSVM method that dynamically adapts the parameter k_{thres}, similar to [98, 99] could be effective.

Multi Stage K-OCSVM

The proposed method uses only the values of the initial alarms in order to filter out those that don't represent an actual attack. That way attacks that cause significant variation in certain features of the OCSVM module are detected, while at the same time other more insidious

attacks are passing undetected. A multi-stage K-OCSVM, both the number of attacks that share common characteristics, like origin, destination, port number etc., and the actual values of the attacks can be developed in order to better detect different kinds of attacks.

Figure 5.2 represents a possible architecture of a two stage K-OCSVM module. The number of the stages can be increased and the alarms produced by each stage can be further aggregated. The fusion of the outputs of the different stages can be done using any of the existing ensemble methods, i.e. majority voting, performance weighting, distribution summation, order statistics etc.

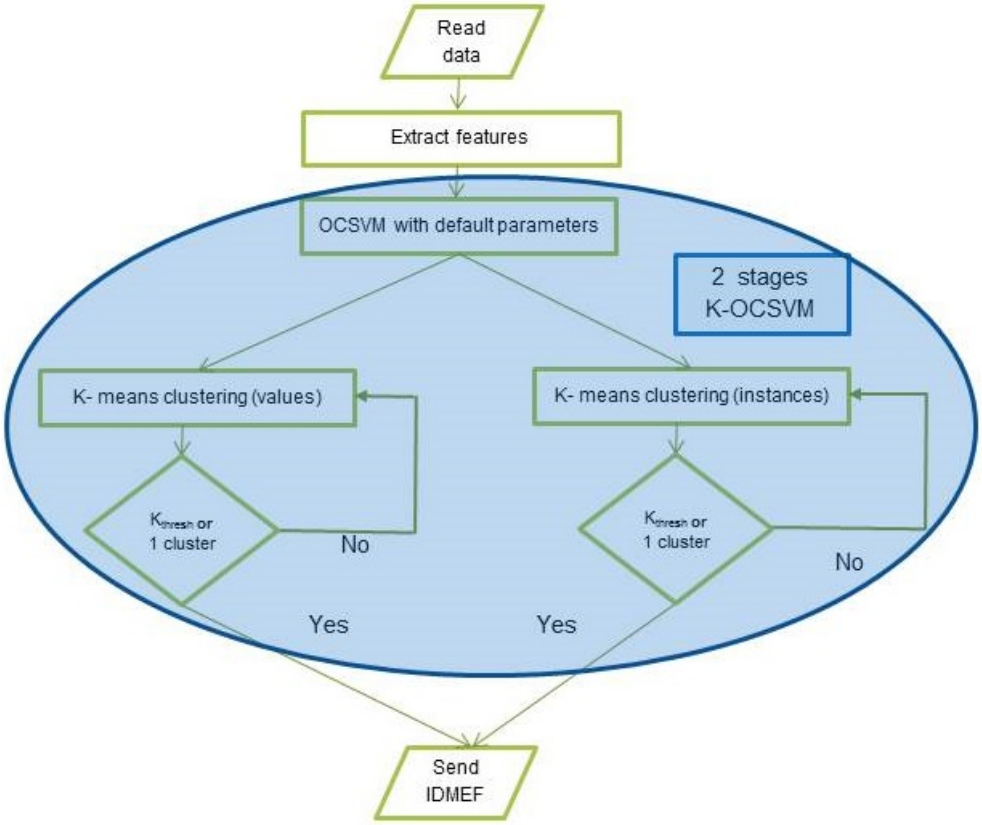

Fig. 5.2 Architecture of a Multi-stage K-OCSVM

The proposed K-OCSVM may not be sufficient to build an effective IDS on its own but is highly valuable when coupled with other methods, especially as an integrated part of a larger framework.

Moreover, it is likely that while anomaly detection provides opportunity to detect unknown attacks it will be necessary to combine them with signature and ruled based IDS components to achieve the greatest accuracy.

Table 5.1 Configurations of the Network

Data	PLC Gr. 1	PLC Gr. 2	HMI 1	HMI 2
1	Active	-	Active	-
2	Active	-	Active	Active
3	Active	Active	Active	-
4	Active	Active	Active	Active
5	-	Active	Active	Active
6	-	Active	Active	-

5.2.2 Future Work, IT-OCSVM

We have established that, relative to IT systems, IC systems (most notably SCADA), have both a greater need for security and a more difficult environment in which to provide it. Support Vector Machines have been shown to be an effective tool for providing intrusion detection, although they lack effectiveness in isolation and are far more powerful when used in conjunction with other methods. None of the materials and research studied examines the adaptability or susceptibility of the proposed IDS systems to changes in the monitored architecture. This is almost certainly due to a large extent to the fact that SCADA systems tend to be relatively static. While Cheung et al [100] argue that model based IDS are suited to SCADA due to their traditionally more stable environment, this stability is decreasing as SCADA systems evolve, undermining this argument.

Based on these findings we conducted a first set of simulation to research how the changes in the network architecture affect the performance of an IDS that is based on the OCSVM principle. The initial simulations showed that there is a weak overall correlation between testing other sets and accuracy (see Figure 5.3). In order to perform the tests, six data sets were provided using the HEDVa. These sets represent network traffic from a SCADA network running in six different configurations. There are two groups of Programmable Logic Controllers (PLCs) and two Human Machine Interface (HMI) devices which may be active on the system. Table 5.1 shows the configuration of the network for each of the data sets:

In order to cope with the drop of accuracy that the IDS demonstrates when the architecture of the system changes, we propose an adaptive mechanism that can be used in any system. A primitive description of this system can be found in [101]. The mechanism that is presented in Figure 5.4 matches the current network traffic of the system to the traffic that the IDS was trained with. Based on the fact that the matching takes additional computation time, the proposed adaptive system imposes a delay on the performance of the IDS. The matching only

Model	Test Data: Self	Test Data: Other
part1	98.76	96.6575
part2	99.5	97.8825
part3	99.31	99.2725
part5	98.53	95.1175
part6	99.15	96.4675
Mean	99.05	97.0795

Fig. 5.3 Mean accuracy

takes into account the different IPs that exist on the network traffic and chooses the OCSVM that was trained to a similar network. This could be extended also to the overall traffic that exists inside the system, based on the fact that the volume of traffic changes between morning and evening and between working days and weekends; especially for a system that controls critical infrastructures that are directly related to human activity, e.g. traffic controls, smart grid etc.

In the core of our adaptive IDS we have included Spearman's rank correlation coefficient [102]. Spearman's rank is a non-parametric measure of correlation widely used to describe the relationship between two variables that is used to report the difference in ranking produced by two methods. Based on this metric we can find the most suitable IDS for the current architecture of the network with a notion of traffic in it, since the metric also ranks the sources based on the total traffic they induce in the system. One important parameter of the proposed Adaptive IDS is how frequent the comparison of the current traffic to the pool of the trained IDS will be and also the number of trained IDSs that we have in order to better match the current situation of the network.

In order to evaluate the efficiency of our proposed Adaptive IDS (A-IDS) we have performed a number of initial tests using the datasets provided from HEDVa. We have tested the proposed A-IDS using different parameters as shown in table 5.2. We investigate how the number of different trained IDSs affect the performance of the system. In our experiments we know the number of different architectures of the system while in a general situation, an automatic method for finding the optimal set of trained IDSs would be needed. Also we investigate how frequent the comparison of current traffic to the ones used for training must

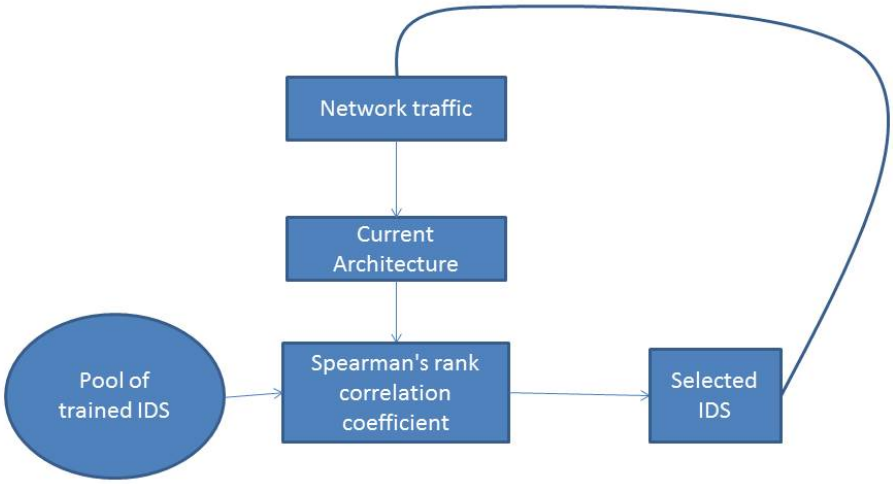

Fig. 5.4 Adaptive IDS (A-IDS)

be, since a small time gap would not include the necessary network traffic sample in order to choose the correct IDS; while on the other hand a big time gap would have, as a consequence, the system to compare possible different architectures.

Table 5.2 Simulation Parameters

Parameters	Range	Default
Number of trained IDS	2 - 5	5
Time gap (min)	1 -10	5

Number of trained IDSs In order to test the efficiency of our proposed A-IDS we have done the following arrangement. The system architecture is initially based on configuration 1 (see Table 5.1 and moves from one configuration to the other following this sequence: Configuration 1, Configuration 2, Configuration 3, Configuration 5 and Configuration 6. During the operation of the system both a static and the proposed dynamic A-IDS are used.

The static IDS is initially trained using Configuration 1 and the dynamic has 3 or 5 trained IDSs, each one representing a different configuration. The system is constantly collecting the network traffic and compares the traffic from the last 5 minutes to the one that was used to train the pool of IDSs using Spearman's rank correlation coefficient. The same experiment is conducted with the system starting from configuration 2, 3, etc. in order to cover all possible

situations. Figure 5.5 shows the aggregated accuracy of the static and the dynamic A-IDSs with 2, 3, 4 and 5 different individual trained IDSs.

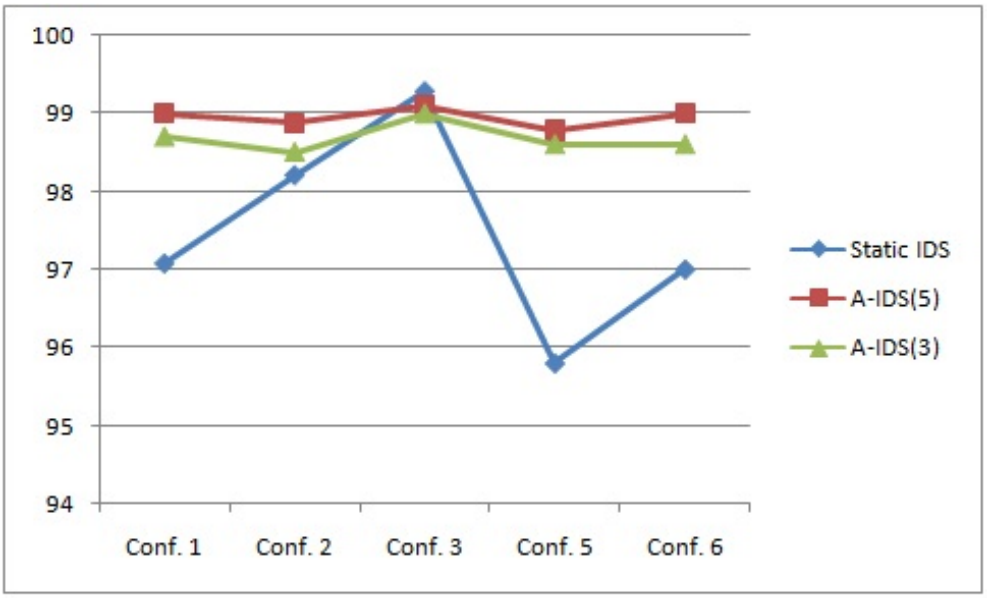

Fig. 5.5 Aggregated accuracy of static and the proposed A-IDS

It is obvious from Figure 5.5 that the A-IDS is more stable in terms of accuracy for all the different simulations conducted and manages to have a stable behavior while the architecture is constantly changing.

Time gap In this subsection we test how much the time gap affects the performance of the proposed A-IDS. As shown in Figure 5.6 when selecting a very small time gap the system is performing worse than the static IDS, due to the fact that most of the time the system chooses a wrongly trained IDS to use. This is caused from the small number of collected packets used for choosing the correct IDS. These small datasets do not represent the actual situation inside the system, causing the A-IDS most of the times to choose wrongly. On the other hand a too big time gap leads to the degradation of the A-IDS to the static one.

These initial tests show that the proposed A-IDS can cope with the architectural changes. Further tests with different datasets and more analytical research of how the different parameters may affect the method are needed. Moreover, the need of new datasets is based on the fact that the evaluation of IDSs using the existing benchmark data sets of KDD99 and NSLKDD does not reflect satisfactory results, as stated in [103]. This is due to three major issues, their lack of modern low footprint attack styles, their lack of modern normal traffic scenarios, and a different distribution of training and testing sets. Recently a new dataset containing malicious data was introduced and analyzed [103]. This data set has 9 types of the modern attacks types and new patterns of normal traffic, and it contains 49 attributes that

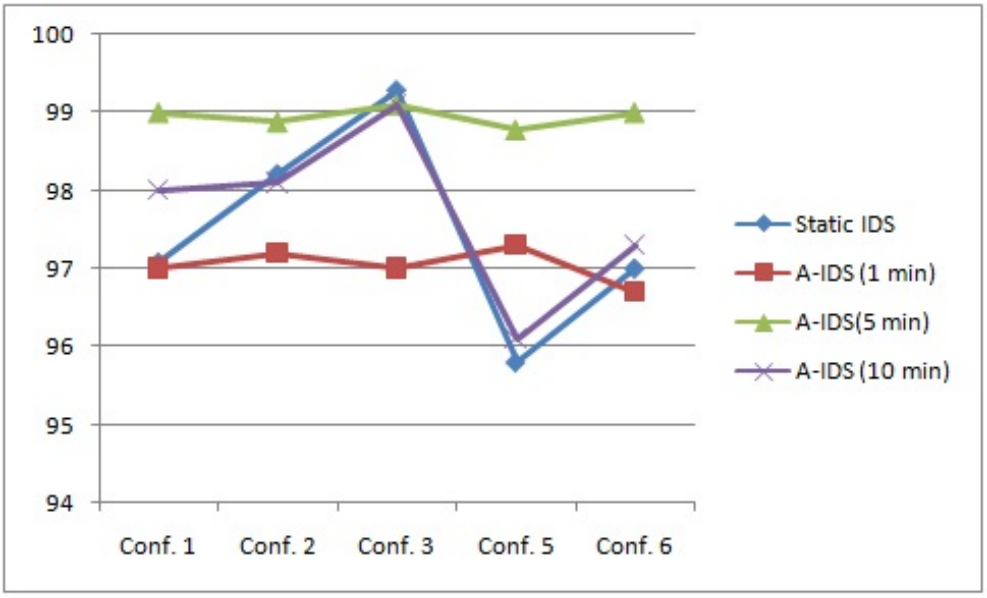

Fig. 5.6 Time Gap affects the performance of the A-IDS

comprise the flow based between hosts and the network packets inspection to discriminate between the observations, either normal or abnormal.

5.3 Discussion

Many IDSs [104, 6] that use network-based auditing study network activity to determine if a node is compromised. This audit can be general (e.g., traffic or frequency analysis) or protocol-specific (e.g., deep packet inspection). The proposed detection mechanisms were parts of a PIDS that was responsible for continuously assessing the electronic security perimeter of the CI. Since only general auditing of network activity (e.g., traffic or frequency analysis) is not always efficient in detecting other attacks, like compromised HMI/PLC or PLC reprogramming attack, other detection mechanisms like snort that perform protocol-specific auditing (e.g., deep packet inspection) were used in parallel. Due to the fact that both methods are based on the OCSVM module, they have the capability of detecting new types of attacks. The key advantage of anomaly-based detection systems are the ability they have to detect zero-day attacks [105]. Having to balance between high accuracy, low false alarm rate, real time communication requirements and low overhead, a combination of several techniques is needed.

The two methods can work on real-time situations without causing high overhead on the medium. The candidate, having worked as a member of the working group that had a

main objective for the adoption of Ethernet Powerlink Standards Group (EPSG) DS 301 – Ethernet Powerlink – Communications Profile Specification as an IEEE Standard, sees that in cases where communication profiles like Ethernet POWERLINK are demanded, the use of any PIDS must be designed in such a way that it can conform with the slot communication management. POWERLINK manages the network traffic in a way that there are dedicated time-slots for isochronous and asynchronous data. It takes care that always only one networked device gains access to the network media. Thus transmission of isochronous and asynchronous data shall not interfere and precise communication timing can be achieved. The mechanism is called Slot Communication Network Management.

Further complication arises because it is known that a large percentage of attacks are induced by inside attackers. Thus perimeter defense alone cannot defend the system. In such cases, the question that one is confronted with is whether there is enough indication of an ongoing attack in the dynamics of the system itself [106]. Despite this range of activities, it has been proven that half of these have human error at their core [107]. Therefore, there should be increased empirical and theoretical research in to human aspects of cyber security based on the volumes of human error related incidents in order to establish ways in which mainstream cyber security practice can benefit.

Security measures tend to neglect that persistent attackers will eventually gain access whatever that perimeter protection may be. One main objective from modern security solutions would be to develop novel methods that could detect and disturb the activities of the attackers once they have gained access inside the system. Special care should be given to the implementation of new strategies that can detect, prevent and mitigate data exfiltration attacks, since intrusion detection/prevention strategies are now deemed to be inadequate for data protection [108].

In order to strengthen the security of SCADA systems, one solution is to deliver defence in depth [109] by layering security controls so as to reduce the risk to the assets being protected. By applying multiple controls “on top of” the information asset (in this case the SCADA and ICS configuration and management data) the architect introduces further barriers, which a threat actor has to overcome. For the more competent threat actors this will slow them down. Within the time it takes to get through some of the controls, the protective monitoring service should have alerted someone to the attack, which will allow further action to be taken (such as dropping the threat actors connection). Defence in depth ensures there is no single point of failure from threats to assets by providing differing barriers (controls) in a layered approach. By applying the industry standard 80/20 rule (80% effectiveness of a control) to each control, it is possible to decrease the likelihood of a compromise exponentially as another control is layered on.

In order to be able to secure a system, analysts must take into account the interdependencies with other systems [110]. A system cannot be safeguarded correctly if treated in isolation. In order to be able a system we must take into account several factors such as security policy, technology, expert knowledge, system dependencies and human factor. The problem of cross-border dependencies in Europe has quite a wide scope. Dependencies exist across borders as regards to network and information security of different services but also between different services across sectors and EU will take specific measures to deal with such dependencies.

SCADA systems are controlling industrial control systems of energy, transport, manufacturing among others. Smart grids are deployed nowadays with main target the improvement of power consumption and the optimization of costs. A smart grid cannot be widely deployed without considering several security and privacy requirements, namely, authentication, integrity, non-repudiation, access control, and privacy [111]. For this reason, recently, researchers in the field of computer security have proposed several privacy-preserving schemes for smart grid communications. Similar to interdependencies for security reasons, interdependent privacy is an open issue for modern SCADA systems.

References

[1] Eric D Knapp and Joel Thomas Langill. *Industrial Network Security: Securing critical infrastructure networks for smart grid, SCADA, and other Industrial Control Systems*. Syngress, 2014.

[2] Adrian Pauna, K Moulinos, Matina Lakka, J May, and T Tryfonas. Can we learn from SCADA security incidents? *White Paper, European Union Agency for Network and Information Security, Heraklion, Crete, Greece*, 2013.

[3] Eric Byres and Justin Lowe. The myths and facts behind cyber security risks for industrial control systems. In *Proceedings of the VDE Kongress*, volume 116, pages 213–218, 2004.

[4] Keith Stouffer, Joe Falco, and Karen Scarfone. Guide to industrial control systems ICS security. *NIST special publication*, 800(82):16–16, 2011.

[5] Javier Lopez, Cristina Alcaraz, and Rodrigo Roman. Smart control of operational threats in control substations. *Computers & Security*, 38:14–27, 2013.

[6] Robert Mitchell and Ing-Ray Chen. A survey of intrusion detection techniques for cyber-physical systems. *ACM Computing Surveys (CSUR)*, 46(4):55, 2014.

[7] Leandros Maglaras, George Drivas, Kleanthis Noou, and Stylianos Rallis. Nis directive: The case of greece. *EAI Transactions on Security and Safety*, 2018.

[8] Yulia Cherdantseva, Pete Burnap, Andrew Blyth, Peter Eden, Kevin Jones, Hugh Soulsby, and Kristan Stoddart. A review of cyber security risk assessment methods for scada systems. *Computers & security*, 56:1–27, 2016.

[9] Leandros A Maglaras, Ki-Hyung Kim, Helge Janicke, Mohamed Amine Ferrag, Stylianos Rallis, Pavlina Fragkou, Athanasios Maglaras, and Tiago J Cruz. Cyber security of critical infrastructures. *ICT Express*, 2018.

[10] Peter Eden, Andrew Blyth, Pete Burnap, Yulia Cherdantseva, Kevin Jones, Hugh Soulsby, and Kristan Stoddart. A cyber forensic taxonomy for scada systems in critical infrastructure. In *International Conference on Critical Information Infrastructures Security*, pages 27–39. Springer, 2015.

[11] Peter Eden, Andrew Blyth, Kevin Jones, Hugh Soulsby, Pete Burnap, Yulia Cherdantseva, and Kristan Stoddart. Scada system forensic analysis within iiot. In *Cybersecurity for Industry 4.0*, pages 73–101. Springer, 2017.

[12] Michael Robinson, Kevin Jones, and Helge Janicke. An introduction to cyber peacekeeping. *arXiv preprint arXiv:1710.09616*, 2017.

[13] Martin Naedele. Addressing IT security for critical control systems. In *System Sciences, 2007. HICSS 2007. 40th Annual Hawaii International Conference*, pages 115–115. IEEE, 2007.

[14] Ehab Al-Shaer and Mohammad Ashiqur Rahman. Smart grids and security challenges. In *Security and Resiliency Analytics for Smart Grids*, pages 3–13. Springer, 2016.

[15] Cristina Alcaraz, Gerardo Fernandez, and Fernando Carvajal. Security aspects of SCADA and DCS environments. In *Critical Infrastructure Protection*, pages 120–149. Springer, 2012.

[16] Scott Dynes. Emergent risks in critical infrastructures. In *International Conference on Critical Infrastructure Protection*, pages 3–16. Springer, 2008.

[17] K.S. Keith Stouffer and Joe Falco. Recommended practise: Improving industrial control systems cybersecurity with defense-in-depth strategies. *Department of Homeland Security, Control Systems Security Program, National Cyber Security division*, 2009.

[18] Cristina Alcaraz, Rodrigo Roman, Pablo Najera, and Javier Lopez. Security of industrial sensor network-based remote substations in the context of the internet of things. *Ad Hoc Networks*, 11(3):1091–1104, 2013.

[19] Grigoris Tzokatziou, Leandros Maglaras, and Helge Janicke. Insecure by design: Using human interface devices to exploit SCADA systems. In *3rd International Symposium for ICS & SCADA Cyber Security Research*, 2015.

[20] Robert Larkin, Juan Lopez, and Jonathan Butts. Evaluation of traditional security solutions in the SCADA environment. In *Proceedings of the 7th International Conference on Information Warfare and Security: ICIW*, page 399. Academic Conferences Limited, 2012.

[21] Eric Byres, P Eng, and ISA Fellow. Using ANSI/ISA-99 standards to improve control system security. *White paper, Tofino Security*, 2012.

[22] Bonnie Zhu, Anthony Joseph, and Shankar Sastry. A taxonomy of cyber attacks on SCADA systems. In *Internet of things (iThings/CPSCom), 2011, 4th International Conference on Cyber, Physical and Social Computing*, pages 380–388. IEEE, 2011.

[23] Tiago Cruz, Luis Rosa, Jorge Proenca, Leandros Maglaras, Matthieu Aubigny, Leonid Lev, Jianmin Jiang, and Paulo Simoes. A cyber security detection framework for supervisory control and data acquisition systems. *IEEE Transactions on Industrial Informatics*, 2016.

[24] Luallen M. Breaches on the rise in control systems: A sans survey 2014. 2014.

[25] Ahmad-Reza Sadeghi, Christian Wachsmann, and Michael Waidner. Security and privacy challenges in industrial internet of things. In *Proceedings of the 52nd Annual Design Automation Conference*, page 54. ACM, 2015.

[26] Eric J Byres, Matthew Franz, and Darrin Miller. The use of attack trees in assessing vulnerabilities in SCADA systems. In *Proceedings of the international infrastructure survivability workshop*. Citeseer, 2004.

[27] Andrew Nicholson, Tim Watson, Peter Norris, Alistair Duffy, and Roy Isbell. A taxonomy of technical attribution techniques for cyber attacks. In *European Conference on Information Warfare and Security*, page 188. Academic Conferences International Limited, 2012.

[28] Tiago Cruz, Jorge Proença, Paulo Simões, Matthieu Aubigny, Moussa Ouedraogo, Antonio Graziano, and Leandros Maglaras. A distributed IDS for industrial control systems. *International Journal of Cyber Warfare and Terrorism (IJCWT)*, 4(2):1–22, 2014.

[29] Robert McMillan. Siemens: Stuxnet worm hit industrial systems. *Computerworld*, 14, 2010.

[30] Florian Skopik. *Collaborative Cyber Threat Intelligence: Detecting and Responding to Advanced Cyber Attacks at the National Level*. CRC Press, 2017.

[31] Sandro Bologna and Roberto Setola. The need to improve local self-awareness in CIP/CIIP. In *Critical Infrastructure Protection, First IEEE International Workshop on*, pages 6–pp. IEEE, 2005.

[32] Ics-cert incident response summary report 2009-2011. 2011.

[33] Prosenjit Sinha, Amine Boukhtouta, Victor Heber Belarde, and Mourad Debbabi. Insights from the analysis of the mariposa botnet. In *Risks and Security of Internet and Systems (CRiSIS), 2010 Fifth International Conference on*, pages 1–9. IEEE, 2010.

[34] Maheshkumar Sabhnani and Gürsel Serpen. Application of machine learning algorithms to kdd intrusion detection dataset within misuse detection context. In *MLMTA*, pages 209–215, 2003.

[35] Anna L Buczak and Erhan Guven. A survey of data mining and machine learning methods for cyber security intrusion detection. *IEEE Communications Surveys & Tutorials*, 18(2):1153–1176, 2016.

[36] Richard Heady, George F Luger, Arthur Maccabe, and Mark Servilla. *The architecture of a network level intrusion detection system*. University of New Mexico. Department of Computer Science. College of Engineering, 1990.

[37] Richard A Kemmerer and Giovanni Vigna. Intrusion detection: a brief history and overview. *Computer*, 35(4):supl27–supl30, 2002.

[38] Julia Allen, Alan Christie, William Fithen, John McHugh, and Jed Pickel. State of the practice of intrusion detection technologies. Technical report, DTIC Document, 2000.

[39] Zibusiso Dewa and Leandros A Maglaras. Data mining and intrusion detection systems. *International Journal of Advanced Computer Science and Applications,*, 7(1):62–71, 2016.

[40] Hervé Debar, Marc Dacier, and Andreas Wespi. Towards a taxonomy of intrusion-detection systems. *Computer Networks*, 31(8):805–822, 1999.

[41] Leandros A Maglaras and Jianmin Jiang. Intrusion detection in SCADA systems using machine learning techniques. In *Science and Information Conference (SAI), 2014*, pages 626–631. IEEE, 2014.

[42] Jinhong Yang, Tingquan Deng, and Ran Sui. An adaptive weighted one-class svm for robust outlier detection. In *Proceedings of the 2015 Chinese Intelligent Systems Conference*, pages 475–484. Springer, 2016.

[43] Hung-Jen Liao, Chun-Hung Richard Lin, Ying-Chih Lin, and Kuang-Yuan Tung. Intrusion detection system: A comprehensive review. *Journal of Network and Computer Applications*, 36(1):16–24, 2013.

[44] Laurens Lemaire, Jorn Lapon, Bart De Decker, and Vincent Naessens. A sysml extension for security analysis of industrial control systems. In *Proceedings of the 2nd International Symposium on ICS & SCADA Cyber Security Research 2014*, pages 1–9. BCS, 2014.

[45] Iñaki Garitano, Roberto Uribeetxeberria, and Urko Zurutuza. A review of SCADA anomaly detection systems. In *Soft Computing Models in Industrial and Environmental Applications, 6th International Conference SOCO 2011*, pages 357–366. Springer, 2011.

[46] Christina Yip Chung, Michael Gertz, and Karl Levitt. Demids: A misuse detection system for database systems. In *Integrity and Internal Control in Information Systems*, pages 159–178. Springer, 2000.

[47] Gisung Kim, Seungmin Lee, and Sehun Kim. A novel hybrid intrusion detection method integrating anomaly detection with misuse detection. *Expert Systems with Applications*, 41(4):1690–1700, 2014.

[48] Mohiuddin Ahmed and Abdun Naser Mahmood. A novel approach for outlier detection and clustering improvement. In *Industrial Electronics and Applications (ICIEA), 2013 8th IEEE conference*, pages 577–582. IEEE, 2013.

[49] Mohiuddin Ahmed and Abdun Naser Mahmood. Network traffic pattern analysis using improved information theoretic co-clustering based collective anomaly detection. In *International Conference on Security and Privacy in Communication Systems*, pages 204–219. Springer, 2014.

[50] Evangelos E Papalexakis, Alex Beutel, and Peter Steenkiste. Network anomaly detection using co-clustering. In *Encyclopedia of Social Network Analysis and Mining*, pages 1054–1068. Springer, 2014.

[51] Wei-Chao Lin, Shih-Wen Ke, and Chih-Fong Tsai. Cann: An intrusion detection system based on combining cluster centers and nearest neighbors. *Knowledge-based Systems*, 78:13–21, 2015.

[52] Mohiuddin Ahmed, Abdun Naser Mahmood, and Jiankun Hu. A survey of network anomaly detection techniques. *Journal of Network and Computer Applications*, 60:19–31, 2016.

[53] Seokjun Lee, Hyunguk Yoo, Jungtaek Seo, and Taeshik Shon. Packet diversity-based anomaly detection system with ocsvm and representative model. In *Internet of Things (iThings) and IEEE Green Computing and Communications (GreenCom) and IEEE Cyber, Physical and Social Computing (CPSCom) and IEEE Smart Data (SmartData), 2016 IEEE International Conference on*, pages 498–503. IEEE, 2016.

[54] Wael Khreich, Babak Khosravifar, Abdelwahab Hamou-Lhadj, and Chamseddine Talhi. An anomaly detection system based on variable n-gram features and one-class SVM. *Information and Software Technology*, 91:186–197, 2017.

[55] Song Han, Miao Xie, Hsiao-Hwa Chen, and Yun Ling. Intrusion detection in cyber-physical systems: Techniques and challenges. *Systems Journal, IEEE*, 8(4):1049–1059, 2014.

[56] IgorNai Fovino, Marcelo Masera, Michele Guglielmi, Andrea Carcano, and Alberto Trombetta. Distributed intrusion detection system for SCADA protocols. In Tyler Moore and Sujeet Shenoi, editors, *Critical Infrastructure Protection IV*, volume 342 of *IFIP Advances in Information and Communication Technology*, pages 95–110. Springer Berlin Heidelberg, 2010.

[57] Yichi Zhang, Lingfeng Wang, Weiqing Sun, Robert Green, and Mansoor Alam. Distributed intrusion detection system in a multi-layer network architecture of smart grids. *IEEE Tran. on Smart Grids*, 2(4):796–808, 2011.

[58] J. Rushi and K. Kang. Detecting anomalies in process control networks. In *Proc. of the 3rd IFIP WG 11. 10 Int. Critical Infrastructure Protection Conference Conference*, pages 151–165. Springer, 2009.

[59] J. Zaddach, L. Bruno, A. Fracillon, and D. Balzarotti. Avatar: A framework to support dynamic security analysis of embedded systems' firmwares. In *Proc. of Net. and Dist. System Security Symposium*, 2014.

[60] R. Karri and J. Rajendran. Trustworthy hardware: Identifying and classifying hardware trojans. *Computer*, 43(10):39–46, 2010.

[61] W. Gao, T. Morris, B. Reaves, and D. Richey. On SCADA control system command and response injection and intrusion detection. In *Proc. of eCrime Researchers Summit*, pages 1–9, 2010.

[62] A. Valdes and S. Cheung. Intrusion monitoring in process control systems. In *Proc. of the 42nd Hawaii International Conference on System Sciences,*, page 17, 2009.

[63] J. Kannan, J. Jung, J. Paxton, and C. Koksal. Semi-automated discovery of application session structure. In *Proc. of the 6th ACM SIGCOMM Conference on Internet Measurement*, page 132, 2006.

[64] Yi Yang, Keiran McLaughlin, Sakir Sezer, Tim Littler, Eul Gyu Im, Bernardi Pranggono, and H.F. Wang. Multiattribute SCADA-specific intrusion detection system for power networks. *IEEE Transactions on Power Delivery*, 29(3):1092–1102, 2014.

[65] B. Zhu. Resilient control and intrusion detection for SCADA systems. Technical report, University of California at Berkeley, 2014.

[66] Leandros A Maglaras, Jianmin Jiang, and Tiago Cruz. Integrated OCSVM mechanism for intrusion detection in SCADA systems. *Electronics Letters*, 50(25):1935–1936, 2014.

[67] Leandros A. Maglaras and Jianmin Jiang. A novel intrusion detection method based on OCSVM and K-means recursive clustering. *EAI Transactions on Security and Safety*, 2(3):1–10, 2015.

[68] Kevin Wong, Craig Dillabaugh, Nabil Seddigh, and Biswajit Nandy. Enhancing suricata intrusion detection system for cyber security in SCADA networks. In *Electrical and Computer Engineering (CCECE), 2017 IEEE 30th Canadian Conference*, pages 1–5. IEEE, 2017.

[69] Imdadul Karim, Quoc-Tuan Vien, Tuan Anh Le, and Glenford Mapp. A comparative experimental design and performance analysis of snort-based intrusion detection system in practical computer networks. *Computers*, 6(1):6, 2017.

[70] Yanxin Wang, Johnny Wong, and Andrew Miner. Anomaly intrusion detection using One Class SVM. In *Information Assurance Workshop, 2004. Proceedings from the Fifth Annual IEEE SMC*, pages 358–364. IEEE, 2004.

[71] Syed Noorulhassan Shirazi, Antonios Gouglidis, Kanza Noor Syeda, Steven Simpson, Andreas Mauthe, Ioannis M. Stephanakis, and David Hutchison. Evaluation of anomaly detection techniques for SCADA communication resilience. In *Resilience Week*, 2016.

[72] Allan Cook, Janicke Helge, Leandros Maglaras, and Richard Smith. An assessment of the application of IT security mechanisms to industrial control systems. *International Journal of Internet Technology and Secured Transactions*, 2017.

[73] Jianmin Jiang and Lasith Yasakethu. Anomaly detection via One Class SVM for protection of SCADA systems. In *Cyber-Enabled Distributed Computing and Knowledge Discovery (CyberC), 2013 International Conference on*, pages 82–88. IEEE, 2013.

[74] J. Ma and S. Perkins. Time-series novelty detection using one-class support vector machines. In *Proceedings of the International Joint Conference on Neural Networks*, volume 3, pages 1741–1745 vol.3, July 2003.

[75] Kun-Lun Li, Hou-Kuan Huang, Shen-Feng Tian, and Wei Xu. Improving one-class SVM for anomaly detection. In *International Conference onMachine Learning and Cybernetics*, volume 5, pages 3077–3081. IEEE, 2003.

[76] Dipankar Dasgupta and Fabio A Gonzalez. An intelligent decision support system for intrusion detection and response. In *Information Assurance in Computer Networks*, pages 1–14. Springer, 2001.

[77] Assaf Glazer, Lindenbaum Michael, and Shaul Markovitch. q-OCSVM: A q-quantile estimator for high-dimensional distributions. In *In Proceedings of The 27th Conference on Neural Information Processing Systems (NIPS-2013),Lake Tahoe, Nevada*, 2013.

[78] Xiaomu Song, Guoliang Fan, and Mahesh Rao. SVM-based data editing for enhanced one-class classification of remotely sensed imagery. *Geoscience and Remote Sensing Letters, IEEE*, 5(2):189–193, 2008.

[79] Eleazar Eskin, Andrew Arnold, Michael Prerau, Leonid Portnoy, and Sal Stolfo. A geometric framework for unsupervised anomaly detection. In *Applications of Data Mining in Computer Security*, pages 77–101. Springer, 2002.

[80] Edwin M Knorr and Raymond T Ng. A unified notion of outliers: Properties and computation. In *KDD*, pages 219–222, 1997.

[81] Xuchun Li, Lei Wang, and Eric Sung. Adaboost with SVM-based component classifiers. *Engineering Applications of Artificial Intelligence*, 21(5):785–795, 2008.

[82] Prabir Burman. A comparative study of ordinary cross-validation, v-fold cross-validation and the repeated learning-testing methods. *Biometrika*, 76(3):503–514, 1989.

[83] Jerome Friedman, Trevor Hastie, and Robert Tibshirani. *The elements of statistical learning*, volume 1. Springer Series in Statistics, Springer, Berlin, 2001.

[84] Eitan Menahem, Lior Rokach, and Yuval Elovici. Combining one-class classifiers via meta learning. In *Proceedings of the 22nd ACM International Conference on Information and Knowledge Management*, pages 2435–2440. ACM, 2013.

[85] Grigorios Tsoumakas, Ioannis Katakis, and Ioannis Vlahavas. Effective voting of heterogeneous classifiers. In *Machine Learning: ECML 2004*, pages 465–476. Springer, 2004.

[86] Myoung-Jong Kim and Dae-Ki Kang. Ensemble with neural networks for bankruptcy prediction. *Expert Systems with Applications*, 37(4):3373–3379, 2010.

[87] Bartosz Krawczyk and Michał Woźniak. Diversity measures for one-class classifier ensembles. *Neurocomputing*, 126:36–44, 2014.

[88] Thomas Philip Runarsson and Magnus Thor Jonsson. Model selection in one-class ν-SVMs using RBF kernels. In *Proceedings of 16th International Congress and Exhibition on Condition Monitoring and Diagnostic Engineering Management*, 2003.

[89] Leandros A Maglaras and Jianmin Jiang. OCSVM model combined with K-means recursive clustering for intrusion detection in SCADA systems. In *Heterogeneous Networking for Quality, Reliability, Security and Robustness (QShine), 2014 10th International Conference on*, pages 133–134. IEEE, 2014.

[90] Leandros Maglaras and Jianmin Jiang. A real time OCSVM intrusion detection module with low overhead for SCADA systems. *International Journal of Advanced Research in Artificial Intelligence*, (3):45–53, 2014.

[91] Grzegorz Kołaczek and Agnieszka Prusiewicz. Social network approach to anomaly detection in network systems. *Intrusion Detection Systems*, 2011.

[92] Prasanta Gogoi, D. K. Bhattacharyya, B. Borah, and Jugal K. Kalita. A survey of outlier detection methods in network anomaly identification. *The Computer Journal*, 54(4):570–588, 2011.

[93] Md Abdul Hamid, Md Shariful Islam, and Choong Seon Hong. Misbehavior detection in wireless mesh networks. In *10th International Conference on Advanced Communication Technology (ICACT)*, volume 2, pages 1167–1169. IEEE, 2008.

[94] Daniel-Ioan Curiac and Constantin Volosencu. Ensemble-based sensing anomaly detection in wireless sensor networks. *Expert Systems with Applications*, 39(10):9087–9096, 2012.

[95] Leandros A Maglaras, Jianmin Jiang, and Tiago J Cruz. Combining ensemble methods and social network metrics for improving accuracy of OCSVM on intrusion detection in SCADA systems. *arXiv preprint arXiv:1507.02825*, 2015.

[96] Aleksandar Lazarevic, Levent Ertöz, Vipin Kumar, Aysel Ozgur, and Jaideep Srivastava. A comparative study of anomaly detection schemes in network intrusion detection. In *SDM*, pages 25–36. SIAM, 2003.

[97] Leandros A Maglaras, Jianmin Jiang, and Tiago Cruz. Integrated OCSVM mechanism for intrusion detection in SCADA systems. *Electronics Letters*, 50(25):1935–1936, 2014.

[98] C Campbell, N Cristianini, and J Shawe-Taylor. Dynamically adapting kernels in support vector machines. *Advances in Neural Information Processing Systems*, 11:204–210, 1999.

[99] Lijuan Cao and Qingming Gu. Dynamic support vector machines for non-stationary time series forecasting. *Intelligent Data Analysis*, 6(1):67–83, 2002.

[100] Steven Cheung, Bruno Dutertre, Martin Fong, Ulf Lindqvist, Keith Skinner, and Alfonso Valdes. Using model-based intrusion detection for SCADA networks. In *Proceedings of the SCADA Security Scientific Symposium*, volume 46, pages 1–12. Citeseer, 2007.

[101] Barnaby Stewart, Luis Rosa, Leandros Maglaras, Tiago J Cruz, Paulo Simoes, and Helge Janicke. Effect of network architecture changes on OCSVM based intrusion detection system. *Industrial Networks and Intelligent Systems, INISCOM 2016. Lecture Notes of the Institute for Computer Sciences, Social Informatics and Telecommunications Engineering*, 88, 2016.

[102] Charles Spearman. The proof and measurement of association between two things. *The American Journal of Psychology*, 15(1):72–101, 1904.

[103] Nour Moustafa and Jill Slay. The evaluation of network anomaly detection systems: Statistical analysis of the unsw-nb15 data set and the comparison with the kdd99 data set. *Information Security Journal: A Global Perspective*, 2016.

[104] Sooyeon Shin, Taekyoung Kwon, Gil-Yong Jo, Youngman Park, and Haekyu Rhy. An experimental study of hierarchical intrusion detection for wireless industrial sensor networks. *IEEE Transactions on Industrial Informatics*, 6(4):744–757, 2010.

[105] Prakash Mandayam Comar, Lei Liu, Sabyasachi Saha, Pang-Ning Tan, and Antonio Nucci. Combining supervised and unsupervised learning for zero-day malware detection. In *INFOCOM Proceedings*, pages 2022–2030. IEEE, 2013.

[106] Sandeep K Shukla. Cyber security of cyber physical systems: Cyber threats and defense of critical infrastructures. In *15th International Conference on Embedded Systems (VLSID)*, pages 30–31. IEEE, 2016.

[107] Mark Evans, Leandros A Maglaras, Ying He, and Helge Janicke. Human behaviour as an aspect of cybersecurity assurance. *Security and Communication Networks*, 9(17):4667–4679, 2016.

[108] Dr Awais Rashid, Rajiv Ramdhany, Matthew Edwards, Sarah Mukisa Kibirige, Ali Babar, David Hutchison, and Ruzanna Chitchyan. Detecting and preventing data exfiltration. *Security Lancaster, Lancaster University*, 2014.

[109] Andy Wood, Ying He, Leandros Maglaras, and Helge Janicke. A security architectural pattern for risk management of industry control systems within critical national infrastructure. *International Journal of Critical Infrastructures*, 2016.

[110] Pete Burnap, Yulia Cherdantseva, Andrew Blyth, Peter Eden, Kevin Jones, Hugh Soulsby, and Kristan Stoddart. Determining and sharing risk data in distributed interdependent systems. *Computer*, 50(4):72–79, 2017.

[111] Mohamed Amine Ferrag, Leandros A Maglaras, Helge Janicke, Jianmin Jiang, and Lei Shu. A systematic review of data protection and privacy preservation schemes for smart grid communications. *Sustainable Cities and Society*, 2018.

Appendix A

Related Publications

The main ideas presented in this thesis appear in the following publications:

1. Tiago Cruz, Luis Rosa, Jorge Proenca, Leandros Maglaras, Matthieu Aubigny, Leonid Lev, Jianmin Jiang, Paulo Simoes, "A Cyber Security Detection Framework for Supervisory Control and Data Acquisition Systems", IEEE Transactions on Industrial Informatics, Vol. 12 Issue 6, pp. 1-11, December 2016, DOI: 10.1109/TII.2016.2599841

2. Leandros A. Maglaras, Jianmin Jiang, Tiago J. Cruz, "Combining ensemble methods and social network metrics for improving accuracy of OCSVM on intrusion detection in SCADA systems", Journal of Information Security and Applications (Elsevier), Vol. 30, pp. 15 - 26, October 2016, DOI: 10.1016/j.jisa.2016.04.002

3. Leandros A. Maglaras, Jianmin Jiang, "A novel intrusion detection method based on OCSVM and K-means recursive clustering", EAI Transactions on Security and Safety, Vol. 2, no 3, e5, pp. 1-10, January 2015, DOI: 10.4108/sesa.2.3.e5

4. Leandros A. Maglaras, Jianmin Jiang, Tiago Cruz, “Integrated OCSVM mechanism for intrusion detection in SCADA systems”, IET Electronics Letters, Volume 50, Issue 25, December 2014, p 1935-1936, DOI: 10.1049/el.2014.2897

5. Leandros A. Maglaras, Jianmin Jiang "OCSVM model combined with K-means clustering for intrusion detection in SCADA systems", Proceedings of the 10th International Conference on Heterogeneous Networking for Quality, Reliability, Security and Robustness (QSHINE 2014), August 2014, pp. 133-134, doi:10.1109/QSHINE.2014.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TII.2016.2599841, IEEE Transactions on Industrial Informatics

A Cyber Security Detection Framework for Supervisory Control and Data Acquisition Systems

***Abstract*—This paper presents a Distributed Intrusion Detection System (DIDS) for Supervisory Control and Data Acquisition (SCADA) Industrial Control Systems (ICS) that was developed for the CockpitCI project. Its architecture was designed to address the specific characteristics and requirements for SCADA cyber security that cannot be adequately fulfilled by techniques from the Information Technology (IT) world, thus requiring a domain-specific approach. DIDS components are described in terms of their functionality, operation, integration and management. Moreover, system evaluation and validation are undertaken within a specially designed hybrid testbed emulating the SCADA system for an electrical distribution grid.**

I. Introduction

Cyber security is currently one of the main concerns for SCADA ICS operators, as a result of a series of recent successful cyber attacks against several targets, such as electric power substations and distribution grids [1], sewage processing units or even nuclear power plants [2]. These have had far reaching impact, affecting a substantial number of persons, potentially causing significant damage and ultimately threatening human lives [3]. Post-event investigation has frequently linked these attacks to the exploitation of vulnerabilities deeply rooted in the ICS design philosophy and related technologies.

The first generations of SCADA systems relied on air gapped isolation, together with a security-by-obscurity approach, based on proprietary or poorly documented technologies to protect the infrastructure. Later on, the introduction of features from the information and communication technologies (ICT) world, such as Ethernet networks and Transmission Control Protocol/Internet Protocol (TCP/IP) stacks, encouraged the interconnection of the ICS and ICT network domains, as well as to the exterior (mostly for remote management) [4].

Increased connectivity, together with the adoption of open and documented protocols exposed the SCADA ICS "black box" to a variety of new threats. Due to their importance, those systems rapidly became popular targets for all sorts of frivolous, ego-driven or vindictive attacks [5]. They have even subject to Advanced Persistent Threats (APTs) operating over extended time periods as part of cyber warfare strategies designed to probe and profile potential targets, with the aim of extracting valuable intelligence data which can be later used to attack civilian or military infrastructures. The Stuxnet [5] worm is such an example, which has raised awareness for the problems of Critical Infrastructure (CI) protection.

A. Motivation and Contributions

While empirical reasoning might suggest otherwise, the two aforementioned domains are significantly different, mainly due to the increasing number of ICT-related technologies assimilated within ICS. ICS operation and design principles follow an order of priorities which places availability and reliability first, followed by information integrity and, finally, confidentiality and security, while ICT follows the exact reverse order [6]. This is considered one of the primary reasons behind SCADA infrastructure security issues, also explaining the need for domain-specific cyber security mechanisms.

Recognizing the need to improve upon existing security solutions for SCADA ICS, the CockpitCI[1] project was set up in order to address the problems of CI interdependence and security in an integrated way. Its objectives included the development and implementation of a cyber security detection framework for SCADA ICS, which is the main subject of this paper. This framework departs from the state-of-the-art in ICS cyber security, by providing a series of benefits, namely:

- **Diversified and innovative probing techniques**, supporting different types of security agents, tuned for each specific ICS domain, together with the development of innovative network, device and process-level security probes, aimed at reinforcing the detection capabilities beyond the existing solutions;
- **Hybrid analysis capabilities**, through integration of both signature and anomaly-based techniques, providing the ability to deal with both known and rogue threats;
- **Integration, scalability and reliability**, adopting a distributed multi-layered design, geared towards scalability and reliability, using message queues to relay normalized events between components.

This solution was also designed with existing infrastructures in mind, by supporting mature technologies and taking advantage of already deployed management and operations support systems to provide topology and asset information.

This paper is organized as follows: the next section will provide an overview of SCADA security, also addressing proposals and initiatives to deal with existing issues. Section 3 presents the CockpitCI cyber detection architecture for SCADA ICS, followed by Sections 4 and 5, detailing the analysis mechanisms and detection agents within the platform, respectively. Section 6 addresses the validation of the proposed solution, the architecture of the testbed used for that purpose, the test methodologies and results. Finally, Section 7 concludes the paper, also presenting directions for future developments.

II. SCADA Security: An Overview

SCADA ICS security encompasses aspects as diverse as the protection of physical infrastructures and processes, communications protocols, asset management or software and hardware

[1]CockpitCI FP7 Project (FP7-SEC-2011-1 285647), http://cockpitci.eu

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TII.2016.2599841, IEEE Transactions on Industrial Informatics

lifecycle management, which cannot be handled in the same way as their ICT counterparts.

As an example, lifecycle management procedures that are commonplace in the ICT world, such as patching and updating a system, require a different approach for ICS. With availability being a top priority (components are frequently kept in operation for years without re-initialization), the impediment/high cost of stopping production in case of (un)planned downtime has prompted some ICS managers to neglect routine updates or patching [7]. Moreover, since software releases are subject to prior certification by equipment manufacturers and solution providers, it might not be possible to deploy an operating system patch or upgrade any other key component, as it has not yet been tested for compliance. This is not even an option for products beyond their support lifespan.

Another distinguishable issue has to do with the deployment of security-related components. Due to the time-sensitive nature of SCADA systems, several equipment and software providers advise against installing anti-virus software in computers hosting Human-Machine Interface (HMI) or server hosts, due to latency concerns. Conventional IT Network Intrusion Detection Systems (NIDS) face similar issues, being considered unsuitable for inline deployment, as they constitute an unwanted point-of-failure, which can also contribute to deteriorate communications latency (a critical aspect, especially for systems dealing with strict, real-time, control).

Also, most mature SCADA protocols, that support the communication between devices, such as Programmable Logic Controllers (PLCs) or Remote Terminal Units (RTUs) and the supervisory stations, are insecure. For instance, the Modbus protocol, which was introduced by Modicon in 1979, is still being widely used and supported for new product lines. Its simplicity, which is one of the main reasons for its popularity, also makes it insecure, as it does not support encryption or any other protection mechanisms. Attempts to update this protocol with encryption or authentication features were unsuccessful, owning to increased hardware and computational overhead on both the PLCs/RTUs [8] and master stations [9], while also being incompatible with existing equipment.

These examples illustrate some of the reasons behind the proliferation of unprotected, unpatched or unsupported systems within SCADA ICS, which are potentially vulnerable to cyber threats. Together with the constraints on the implementation of counter-measures [10] this situation has made evident the need for a domain-specific approach for ICS security, a subject next discussed.

A. *Towards cyber security awareness*

At first, several SCADA operators tended to overlook security issues, mainly because it was deemed too costly and complex to update entire infrastructures, just to protect them from seemingly far-fetched threats. This applied all the more, when considering the potential implications of these updates, such as going through planned downtime procedures and further expenditure or abandoning proven and mature platforms, valued for their reliability.

Once cyber security awareness started to grow among the ICS community (a situation partly owing to the increase of ICS-related security incidents), organizations in industry, researchers and governments began working towards improving security in SCADA systems. For example, the National Institute of Standards and Technology (NIST) has published a reference document [11] that provides an overview of ICS threats and vulnerabilities, recommending adequate countermeasures and policies. The Industry Standards Organization (ISA) also published similar guidelines [6], as well as other organizations such as the North American Energy Reliability Committee (NERC) and the International Electrotechnical Commission (IEC), which have published recommendations for infrastructure protection for electric production and distribution [12], as well as the U.S. Nuclear Regulatory Committee (USNRC) regulations for nuclear infrastructures [13], just to mention a few. ICS cyber security issues were also addressed as part of wide scale-efforts developed within European FP7 projects such as CRISALIS, PRECYSE and VIKING [10].

There has also been considerable amount of work regarding SCADA intrusion and anomaly detection, including Intrusion Detection Systems (IDS) [14], device-level anomaly detection and classification [15], IDS solutions combining network traces and physical process control data [16], detection based on traffic and protocol models [17] and approaches based on machine-learning techniques [18], among others. Acknowledging the fact that each detection technique has its own merits and scope, the CockpitCI architecture was built on the idea that the whole is greater than the sum of the parts, by bringing several of these techniques together within an inclusive detection framework, which is described next.

III. The CockpitCI Distributed IDS Architecture

One of the primary goals of the CockpitCI project was building effective cyber detection capabilities for SCADA ICS, integrated within a Dynamic Perimeter Intrusion Detection System (PIDS), which would be responsible for continuously assessing the electronic security perimeter of the CI.

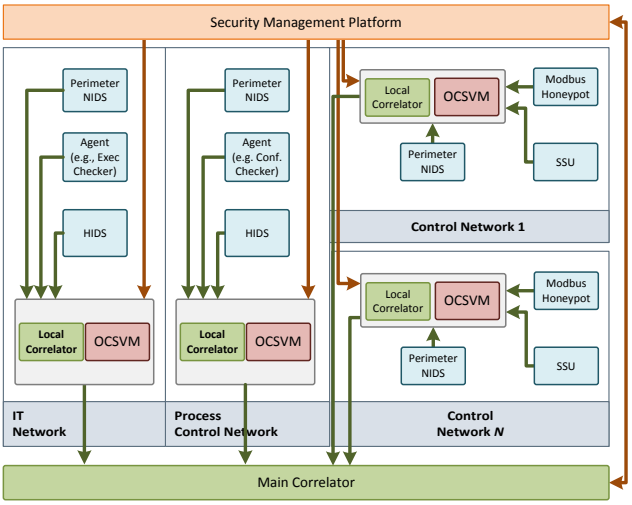

Figure 1. High-level view of the CockpitCI PIDS architecture

The PIDS (see Figure 1) is organized along the distinct network domains of the CI, each one with a well-defined

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TII.2016.2599841, IEEE Transactions on Industrial Informatics

security scope: Control, Process Control, and IT (information technology). It is able to enforce security policies customized to the particular characteristics of each domain, each having its own analysis mechanisms, fed by several detection agents. This architecture was designed to integrate different detection strategies, namely:

- **Detection agents**, comprising specialized network probes (such as NIDS), domain-specific honeypots [19] , host-level agents (using HIDS–Host IDS and other agents) and specialized device monitoring components;
- **A distributed multi-zone, multi-level correlation structure**, processing the event feed from the detection sensor layer, complemented by **machine-learning capabilities**, in the form of One-Class Support Vector Machine (OCSVM) adaptive anomaly detection modules;
- **Usage of topology and system-specific detection mechanisms**, due to the fact that the role, placement and behaviour of ICS components tend to remain unchanged over larger time spans than their ICT counterparts. Therefore, the analysis layer is also provided with information from topology databases or asset management systems.

The PIDS operation is orchestrated and managed by means of a Security Management Platform (SMP – see Figure 2), which oversees the maintenance and management of detection agents and analysis components (correlators, OCSVM analysis), whilst also performing regular in-place security auditing of the protected infrastructure. The SMP adjusts the detection agent sensitivity threshold according to the overall risk level of the infrastructure (adjusted to the estimated probability of an attack event occurring) and also feeds other CockpitCI system components with information from the detection layer, through a secure mediation gateway.

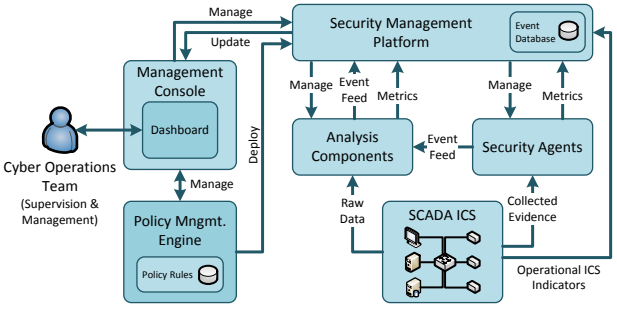

Figure 2. CockpitCI PIDS orchestration

Event messages exchanged between PIDS components are encoded using the Intrusion Detection Message Exchange Format (IDMEF - RFC4765)[2], which establishes a neutral format for intrusion detection related events. Also, the baseline IDMEF schema is encoded using the Extensible Markup Language (XML), allowing it to be updated or customized without disrupting compatibility.

IDMEF event messages generated by detection agents within each domain are published to a broker that routes them to a message queue subscribed by a corresponding local correlator. Once correlation is finished, each local correlator publishes the results to a different queue subscribed by the main correlator - the same queue is also fed by OCSVM components. Finally, the events generated by the main correlator are queued to be sent to the SMP. Each queue (an "Event Bus") is implemented using the Advanced Message Queuing Protocol (AMQP), which defines a neutral (IDMEF-compatible) message encoding method, whilst also providing secure and reliable transport with high-availability features.

[2]H. Debar, D. Curry, and B. Feinstein. (2007) The intrusion detection message exchange format

IV. Analysis components

The PIDS analysis capabilities are provided by two types of components: the correlators and OCSVM modules. Together, they bring the benefits of both signature-based and anomaly detection techniques within the cyber detection framework. This section analyzes these mechanisms in detail.

A. IT-OCSVM

When designing an IDS for ICS, developers typically have to face a difficult situation in that it is not easy to find datasets that contain malicious traffic in order to train a typical machine learning algorithm properly. On the other hand, the datasets that can be collected from a real Industrial network may contain traffic that is not legitimate, while the system is agnostic of this In order to overcome these issues, the One-Class SVM (OCSVM) has become a widely applied method for outlier detection [20]. It can be trained with datasets that contain only legitimate traffic. Once the model is properly trained, it can detect any misbehavior of the system that is due to possible attacks. The OCSVM classification technique can detect novel attacks, as long as they produce network traffic, and is capable of performing real time classification, due to its low computational time.

The OCSVM method has some drawbacks though, like production of high false positives, difficulty to cope with different nature of attacks, induction of high overhead and the inability to classify the attacks. All these issues reveal the need for the creation of more sophisticated detection methods [21] for SCADA systems. Based on these findings we have developed and validated a new integrated detection mechanism, namely the IT-OCSVM [18].

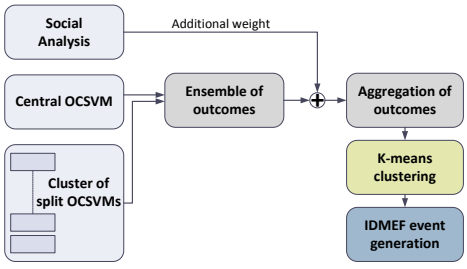

Figure 3. Architecture of the IT-OCSVM detection mechanism

The novel IT-OCSVM method is capable of performing outlier detection with high accuracy and low overhead within a temporal window, adequate for the nature of SCADA systems. In order to achieve these goals, the proposed method consists

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TII.2016.2599841, IEEE Transactions on Industrial Informatics

of several distinct stages that need to be carried out, as shown in Figure 3. The first, similar to any other SVM methods, consists of choosing the most appropriate features for the training of the OCSVM models and the pre-processing of raw input data. In IT-OCSVM alongside the central OCSVM a cluster of split OCSVM models is created.

In the core of the novel IT-OCSVM mechanism lies the OCSVM algorithm which is used in order to map the input data, the network traffic, into a high dimensional feature space (using a Kernel) and through iterations tries to find the maximal margin hyperplane in order to separate the training data from the origin in an optimum way. The OCSVM can be viewed as a two class SVM where all the training data lie on the one class and the origin is the only member of the second class. The hyperplane that is created using the following classification rule:

$$f(x) =< w, x > +b \tag{1}$$

where w is the normal vector and b is a bias term. By solving an optimization problem, OCSVM tries to find the rule f that creates the maximal margin between the origin and the training data. Using the same rule on real time test data x we can assign a label $f(x)$, which categorizes the data as anomaly when the value of $f(x)$ is negative. On the IT-OCSVM detection agent several OCSVM algorithms run in parallel on separate training datasets, labeling the data as normal or anomalous.

The next stage is the assembling of the information gathered by the discrete OCSVMs. The outcomes of both the central OCSVM and the split OCSVMs are combined by using a simple mean value method. They are then given additional weights through the use of Social Network Analysis (SNA) metrics (Equation 2).

$$q_e(i,j) = \sum_{n=1}^{N} w_i d_t(i,j) \tag{2}$$

where, $d_t(i,j)$ is the outcome of each individual classifier n for sample data i that originates from node j and w_i is the weight given to each classifier.

More specific, Spearman's rank correlation coefficient is used in order to add weight to alerts produced from different sources based on the protocols that each source uses during the training and the testing phase (Equation 3).

$$p = 1 - \frac{6\sum d_i^2}{n(n^2-1)} \tag{3}$$

The new outcomes are further aggregated and classified into categories, using a two-stage K-means clustering algorithm. During the first stage of fusion individual alerts are grouped together according to their origin. Each alert i has an initial value q_i (and a source j) based on the procedures described above. Using the equations from (4), each aggregated alert j is assigned two values qa_j and qb_j that represent its severity. This severity comes from both the sum of the values of the initial alerts and the number of attacks that originate from the same node, thus indicating that this node probably misbehaves.

$$qa_j = \sum_i q_s(i,j), \qquad qb_j = \sum_i 1, \forall\ q_s(i,j)\ with\ source\ j \tag{4}$$

During the second stage of fusion K-means clustering is used in order to divide the alarms into: possible, medium and severe (See Equations 5 and 6).

$$SSE = \sum_{k=1}^{K}\sum_{j=1}^{N_k} ||qa_j - \mu_k||^2 \tag{5}$$

where, N_k is the number of instances belonging to cluster k and μ_k is the mean of cluster k, calculated as the mean of all the instances belonging to the cluster i

$$\mu_{k,i} = \frac{1}{N_k}\sum_{q=1}^{N_k} qa_q, j\ \forall j \tag{6}$$

We use two k-means algorithms that run in parallel and each alarm is assigned a final classification: possible, medium or severe. These alarms are sent to the correlator using IDMEF messages containing information such as the nature of the alert, the location of the attacker in the system and the time of detection. The proposed IT-OCSVM analysis mechanism runs in a distributed way and can be used in any SCADA system after proper training. The performance of the mechanism is enhanced by the combination of SNA metrics with machine learning techniques and can be easily incorporated into a soft real time detection system, due to its low overhead.

B. Correlation

As shown in Figure 1, local correlators provide the first event processing stage, being fed by the detection agents deployed within their network domain and sending their output to the main correlator in the form of IDMEF messages. Their role is to filter, process and relate security events within a network domain, also delivering noise filtering and event reduction capabilities, aggregating alerts produced by several detection agents or multiple events from a single source. This approach provides context separation, whilst at the same time allowing for increased scalability and efficiency, as the main correlator can concentrate on multi-step, attack focus recognition correlation, as well as alert prioritization.

The generic PIDS correlator architecture (see Figure 4) encompasses two event interfaces: one to receive events (which may arrive from detection agents, on local correlators or from the local correlators, on the main correlator), and another to send resulting events (to the main correlator, on local correlators or to the SMP, on the main correlator), both using an AMQP-based Event Bus.

The correlator components use the Esper Complex Event Processing (CEP) tool, a flexible and mature event processor, which exhibits a good equilibrium between CPU, memory usage and execution speed in demanding situations. It natively accepts XML-encoded events, which is convenient as IDMEF is an XML-based format. Moreover, it is able to perform correlation over sliding time windows using several data sources,

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TII.2016.2599841, IEEE Transactions on Industrial Informatics

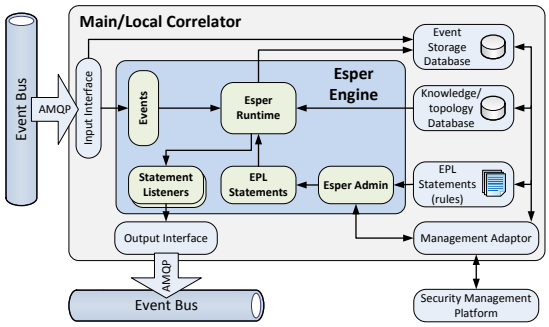

Figure 4. CockpitCI generic correlator architecture

such as input events, persisted and cached information as well as external data sources such as asset management platforms and topology databases. Correlation rules are encoded using Event Processing Language (EPL) statements.

The same architecture is used for both the local and main correlators. Not only does this provide design uniformity, but it also eases management and maintenance, as all correlators share the same rule syntax, thus allowing for rules to be reused across different contexts.

V. Detection Agents

Detection agents constitute the lowest layer of the PIDS architecture, whose role is to collect evidence of suspicious activity directly from the ICS infrastructure elements used for security analysis. They gather information, normalizing and feeding it to the local correlator for its network domain. The PIDS includes a diversity of security probes and detection agents, including (but not restricted to) the following:

- **Network IDS:** each network domain perimeter is monitored using separate NIDS instances for IT, Process Control, and Control Networks; feeding security events to the respective zone correlator. Snort was used for this purpose, but other NIDS can be integrated;
- **Host IDS:** Host IDS are deployed in the ICS stations/servers, providing host-level anomalous behaviour detection. OSSEC was adopted for this purpose, although different HIDS can also be used;
- **Honeypots:** these components act as decoys, to lure and uncover attackers. The PIDS encompasses three types of honeypots, one for each ICS network domain [22] ;
- **Shadow Security Unit:** a security and safety device that is able to monitor a PLC or RTU continuously [23];
- **Exec Checker:** by passively intercepting network traffic, the Exec Checker is able to capture and reassemble a suspicious binary file, in order for it to be analyzed;
- **Output Traffic Controls:** this tool analyses network usage profiles in order to detect the presence of Remote Access Trojans, regularly scanning system components to search for known toolbox signatures;
- **Configuration Checker:** this tool implements system configuration monitoring, checking for unauthorized modification or corruption of configuration data sources.
- **Behavior checker:** this tool monitors and analyzes the behavior of low-level software and hardware indicators, such as temperature and processor activity, in order to avert or anticipate accidental or intentional disruption.

All detection agents publish IDMEF security events on the Event Bus used by its local correlator and since each detection tool has its own eventing format, a generic embedded adaptor was developed to perform normalization. A uniform interface provides agent management capabilities, via the SMP.

Among these detection agents, the Shadow Security Unit and the SCADA Honeypot deserve particular attention, due to their innovative nature. Both will be next analyzed in detail.

A. Shadow Security Unit

The Shadow Security Unit (SSU) [23] was developed to address the protection of RTU and PLC devices within existing SCADA systems. This was deemed a critical PIDS requisite, due to reasons such as the need to deal with zero-day vulnerabilities or because mature device families with a poor security record often have a large deployed base, the updating or replacement of which is considered unfeasible.

The SSU (see Figure 5) is a device attached in parallel to RTU/PLCs that is able to intercept and decode inflight SCADA Protocol Data Units (PDU), correlating extracted data with the state of the physical I/O channels interfacing with field actuators and sensors. It is a security and safety probe, providing continuous behavior and health device monitoring.

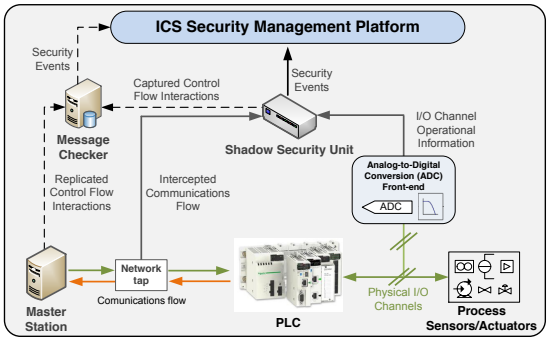

Figure 5. The Shadow Security Unit

The SSU requires minimal modification of the existing topology, being deployed out of the critical control path. This reduces the possibility of interfering with the operation of the protected device due to an eventual malfunction. Besides behavior auditing, the SSU incorporates capabilities such as command stream interception and processing, continuous network flow monitoring and message integrity checks. An optional Message Checker system (see Figure 5) can be used to implement a closed loop between the SCADA stations and the PLCs for communications integrity checking (for instance, to ensure end-to-end SCADA PDU integrity).

B. SCADA Honeypot

One of the innovations of the CockpitCI PIDS is the SCADA honeypot [22], conceived for deployment in the process control network of a SCADA ICS, operating side-by-side with the PLCs, RTUs and other devices, using the network's unallocated IP addresses.

The SCADA honeypot constitutes a decoy for potential attackers, attracting their attention and reporting any suspicious interactions. This is achieved by emulating both the functionality and service set of a commercial PLC, with the purpose of engaging the attacker in intensive probing activities, whose traces can be used for profiling purposes. Its components comprise a fully functional Modbus TCP emulator, including register banks and functions, further complemented by other protocol modules providing file transfer and management services, which are commonplace in commercial PLCs, as well as network portscan detection capabilities.

VI. Deployment and Evaluation

This section details the PIDS validation process, comprising the testbed architecture, IT-OCSVM performance, functional validation use cases, and the event processing capacity tests undertaken to estimate the scalability of the solution.

A. IT-OCSVM initial validation

In order to validate the performance of the IT-OCSVM mechanism that is a core part of the PIDS, a small-scale testbed was created, providing the means to mimic a SCADA system operating both in normal conditions and under the influence of cyberattacks. The testbed architecture was comprised by process control and control network elements, which included a HMI station (for process monitoring), a managed switch (with port monitoring capabilities for network traffic capture), and two PLC units, for process control.

Tests evaluated the performance of the IT-OCSVM mechanism using three types of attacks (later described into more detail in this chapter), namely: network scan, Man-in-the-Middle (MITM) using ARP spoofing and Denial-of-Service (DoS). In all three scenarios the IT-OCSVM module was fed with network traffic and performed data analysis in order to detect malicious behavior. The testing data consisted of normal and attack data and the composition of the data sets is as follows:

- Testing set-A' : Normal data
- Testing set-B' : **ARP spoofing** attack + **Network scan**
- Testing set-C' : **Flooding DoS attack + Network scan**
- Testing set-D' : **MITM attack**

The proposed IT-OCSVM detection mechanism exhibited adequate performance in all attack scenarios, while managing to achieve both high detection accuracy (DA) and low false alarm rates (FAR), as shown on Table I.

Table I
Performance evaluation of the IT-OCSVM module

Dataset	*DA*	*FAR*
A	98.81%	1.18%
B	94.6%	3.25%
C	95.20%	1.51%
D	96.37%	2.3%
All	96.3%	2.5%

Based on the performance exhibited by the IT-OCSVM [24] module, it was decided to deploy it along with the rest of the PIDS components on the platform validation testbed, described on the following subsection.

B. The HEDVa testbed

The Hybrid Environment for Development and Validation (HEDVa) was designed by the Israeli Electric Company for the development and validation of CIs as well as security research projects, being used to implement the attack use cases for the PIDS validation effort. Its architecture (see Figure 6) constitutes a hybrid approach, because of the existence of real/physical SCADA and network/telecom infrastructure components, which are used to implement a simulation model of the electric grid elements.

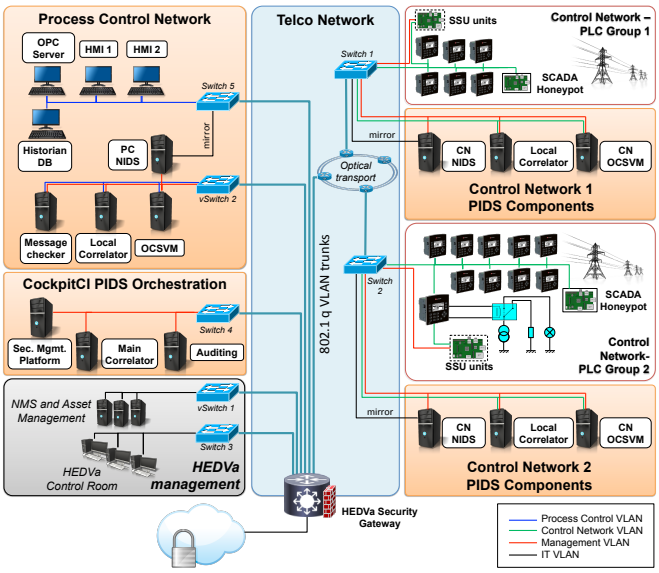

Figure 6. HEDVa networking architecture

The CockpitCI emulated HEDVa scenario corresponds to an energy distribution grid, as shown in Figure 7. Its implementation required the development and testing of different modeling techniques, also including related key performance indicators, fine-tuned with data from production scenarios. Grid elements such as breakers and feeders are controlled by real PLCs in a simulation model, with voltage and current values for segments being calculated accordingly using a mathematical model of the physical grid. In case of failure, the scenario is reconfigured accordingly with the operator's Fault Isolation and Service Restoration procedures.

C. Functional validation

This subsection presents the most relevant use cases for PIDS validation, which were implemented on the HEDVa.

*1) **Layer 2/3/4 attacks**:* These scenarios correspond to malicious activities targeting the SCADA network resources associated to the data-link (2), network (3) and transport layers (4) of the OSI model. For this category, two validation use cases were chosen: a FIN/SYN Scan and a SYN Flooding.

A **port scan** is a typical first step for an attacker to obtain information about the network's hosts and its topology, using

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TII.2016.2599841, IEEE Transactions on Industrial Informatics

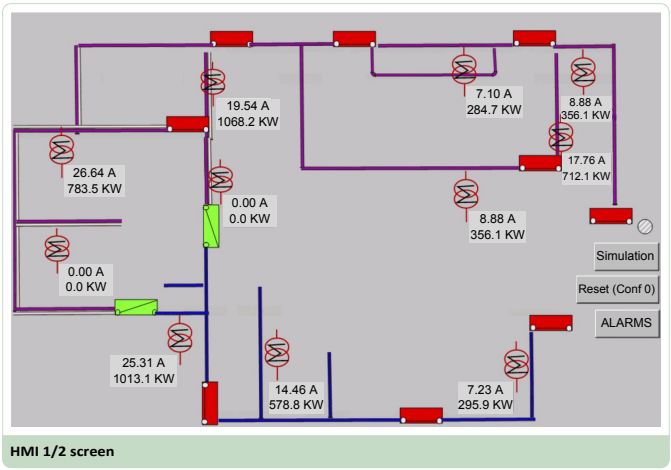

Figure 7. HEDVa grid scenario, with breakers and substation feeders

techniques such as SYN, ACK or FIN scans (among others). This use case was validated on the HEDVa using the *nmap* tool to send a series of SYN and FIN probes (see Figure 8) to devices on a network scope, to scan for open TCP/IP ports.

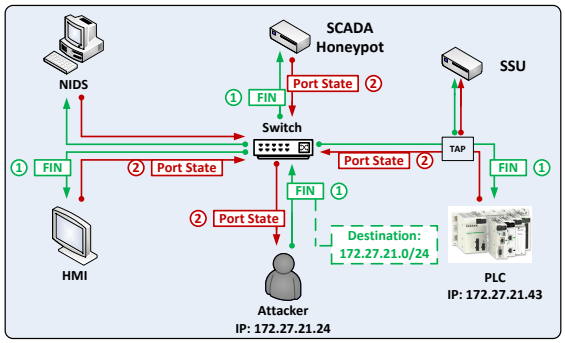

Figure 8. TCP FIN scan

The **SYN flooding** Denial-of-Service (DoS) attack abuses TCP connections, by sending a high volume of SYN packets to the PLC/RTU. This creates a large number of half-open connections on the target device, leading to resource exhaustion and the inability to answer new requests. For validation purposes, SYN flood attacks were implemented using the *hping* tool, against a PLC and an HMI, causing loss of process visibility.

Functional validation demonstrated the effectiveness of several PIDS components against both use cases:

- the NIDS and the OCSVM modules, which detect the corresponding abnormal traffic patterns;
- the SSU authorized device validation and communications stream analysis modules, that detect the anomalous traffic patterns, over a time window (see Figure 8);
- the SCADA Honeypot, which flags up abnormal interactions.

Moreover, in the event that the FIN/SYN packets are forged from a legitimate host, it becomes possible to detect the attack, because of the SSU auto-similar flow pattern analysis features.

*2) **SCADA protocol and process-level attacks**:* SCADA protocols can be specifically targeted by corruption, interception or tampering attacks. These can potentially lead to a loss of confidentiality, visibility, or device connectivity, also providing support to implement process-aware attacks, thus compromising safety and security. A **Man-in-the-Middle (MITM)** attack using Address Resolution Protocol (ARP) poisoning was chosen as the validation use case for this category, exercising protocol and process-level detection capabilities.

The MITM use case starts with an intruder host within the HEDVa Control Network sending unrequested ARP replies to poison the targets' ARP caches and hijack the Modbus TCP protocol flows between the HMI 1 system and the PLCs; this allows for the attacker to eavesdrop communications and gather information about the SCADA network and the process. Afterwards, the attacker will use the previously gathered information to craft a scripted real-time simulator using the *Scapy* tool (see Figure 9 - green flows), reproducing the normal operation of the simulated grid and providing nominal status data back to the HMI 1, while directly manipulating the PLCs, without any visible supervisory feedback.

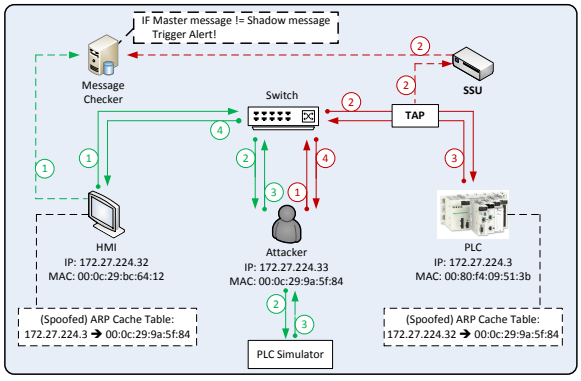

Figure 9. ARP-based, Man-in-the-Middle attack against PLC+SSU

Because the HEDVa grid model depends on the coordinated operation of all PLCs, the attacker scripts are able to respond to HMI operations and react accordingly, simulating the correct reaction from the PLCs – as such, when the HMI operator operates a specific breaker (managed by a PLC), the entire view is updated with the correct energy values of affected segments. This requires the attacker to properly handle all the TCP and Modbus protocol semantics in real time, in order to keep communication integrity and remain undisclosed.

Functional validation demonstrated the capability of several PIDS components for detecting the MITM attack, namely:

- the NIDS, which detect network traffic coming from a station not registered in the asset management or topology databases, together with an anomalous ARP reply traffic pattern (inter-arrival times below normal thresholds);
- the Control Network OCSVM module, that detects an ARP packet rate above the classification threshold;
- the SCADA Honeypot, which detects unrequested ARP replies on the Control Network domain;
- the SSU station validation and high-level command flow processing modules, which detect Modbus command pat-

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TII.2016.2599841, IEEE Transactions on Industrial Informatics

terns deviating from normal sequences (modeled using Markov chains), eventually involving unknown peers;
- the SSU message checker, that detects inconsistencies between the HMI 1 interactions and Modbus commands arriving at the PLC (see Figure 9).

*3) **Compromised HMI/PLC**:* This scenario corresponds to a situation in which an HMI or a PLC becomes compromised. For functional validation purposes, two cases were considered: an HMI infected with malicious software and a PLC reprogrammed by an attacker to disrupt a controlled process.

The **HMI malware infection attack** was performed using the *metasploit* tool to inject a custom malware payload, abusing an Operating System vulnerability allowing for remote execution of binaries. This exploit was used to modify a series of Basic Control Engine script files from the Cimplicity HMI software to hide malicious activity, tampering with the events used to update the HMI, while directly manipulating PLCs.

After the malware infection, the HIDS detected a change on HMI file set signatures, together with an unknown process, while the Exec Checker was able to detect the presence of an executable payload in transit for HMI 1. This was further confirmed by the presence of anomalous traffic patterns after the malware infection, because of Command and Control interaction (reported from the NIDS, OCSVM and output traffic controls). Moreover, the behaviour analysis and command interception features of the SSU detected anomalous command patterns, despite the fact that they were coming from a legitimate source. The SMP security auditing module could have prevented this attack, since it detected the exploited vulnerability, which was left unpatched for test purposes.

The **PLC reprogramming attack** was implemented using a vulnerability that allows for unauthorized upload and download of ladder logic code on Schneider PLCs [25]. This was used to reprogram one of the PLCs to flip the state of a breaker at a rate fast enough to cause damage, while hiding its actions, by providing nominal state information to the HMI.

While the abuse of function code 90 (0x5a) can be detected on the NIDS (via signature rules or unauthorized flow detection), there is a potential contention problem if the switch monitor port where the NIDS is connected receives traffic from other interfaces, all running at the same speed – this may cause packet loss on the NIDS port. For this specific attack use case, the SSU provides two effective mechanisms: command stream interception (which captures the anomalous interaction) and also behaviour checking, which compares the behaviour of the PLC (command interactions and physical I/O ports) against a model of its nominal operation mode, enabling it to detect an anomalous switching rate on a physical I/O channel. Further information about the performance of the PIDS can be found on the public deliverables of the CockpitCI project [3] [4].

[3]Deliverable 5.4 CockpitCI System Factory Trials Report, https://www.cockpitci.eu/wp-content/uploads/2015/04/CockpitCI-D5.4-CockpitCI-System-Factory-Trials-Report.pdf

[4]Deliverable 6.1 – Validation Plan-Final , https://www.cockpitci.eu/wp-content/uploads/2015/04/CockpitCI-D6.1-Validation-Plan-Final.pdf

D. Event processing capacity evaluation

To evaluate the capacity of the event transport and correlator components, a series of off-band performance tests were executed on a validation testbed (see Figure 10), composed of three VMWare ESXi 5.5 virtual machines running the CentOS 6.4 Linux OS: an event publisher (simulating a PIDS agent), an event processing node (with the RabbitMQ broker and the PIDS correlator), and a consumer application for events.

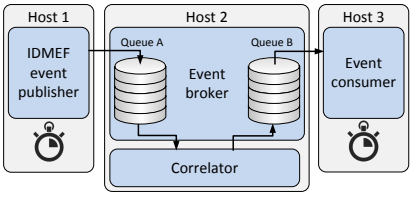

Figure 10. Testbed for event processing capacity tests

The VM host is equipped with an Intel Xeon X5660 6-core CPU running at 2.80GHz, and 32GB RAM. Core and resource affinity parameters were tuned to avoid contention: VMs for the event publisher and consumer used 1 core and 4GB RAM each, while the event processing host was configured with 2 processor cores and 6GB RAM - all VMs have equal priority I/O shares, for a non-SSD datastore. Network connectivity is provided using a virtual switch running at host speed, reducing the physical network latency and speed overhead issues.

The event processing node consumes events from queue A on the broker and then, after processing, publishes results to queue B. The correlator was configured with 20 statement/rules, for testing purposes - all received events trigger a rule and there is no second order correlation. For timing purposes, the publisher records a timestamp on the IDMEF messages, enabling the consumer to assess processing delay by calculating the delta relative to the reception time of the corresponding resulting event. System clocks on all hosts are synchronized using the Network Time Protocol (NTP) .

Validation results will be next presented, with a focus on event processing performance and CPU/memory overhead.

*1) **Event processing performance**:* The event processing capacity tests for the correlation component were executed using IDMEF event sizes ranging from 1 (minimum size payload, with mandatory attributes) to 20 KB (corresponding to a complex meta-event with several references). Tests also evaluate the overhead for the queue acknowledgment mode, which can either be enabled, so that the publisher receives confirmation that the message was received by the consumer or persisted to disk by the broker, or in *AutoAck* mode, in which case the broker issues an acknowledgment as soon it receives a message. Encryption overhead was also measured.

Figure 11 shows average results for 15 test rounds (standard deviation between 4 and 11%) of 10.000 event bursts, executed with persistence support so that the broker saves messages to mass storage when they cannot be consumed right away, allowing reliable delivery, albeit with a performance penalty. The results show the event processing rate to decrease with the message size, as larger messages take longer to send and marshall/unmarshall. Also, confirmation and encryption over-

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TII.2016.2599841, IEEE Transactions on Industrial Informatics

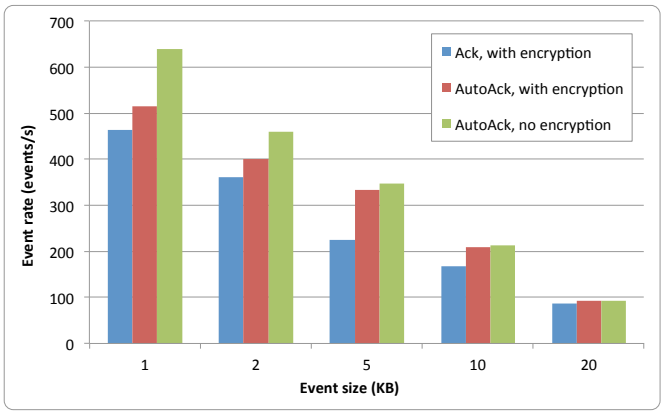

Figure 11. Event processing rate

head decrease along with the message size, with the former having a bigger (albeit increasingly insignificant) impact, when compared with the latter, especially for event sizes between 5 and 20KB. Results were considered adequate for PIDS usage.

*2) **CPU and memory overhead**:* CPU usage for Host 2 was continuously monitored during the tests. For 1, 2 and 5KB event bursts, the CPU capacity of the host was exhausted, a situation that is mainly attributed to the correlator, which alone takes an average 70% of total CPU time for each test case – for 10KB and 20KB, total CPU usage averaged at 82 and 80% respectively. These results can be improved: a physical networking infrastructure will have a mitigating effect on event bursts; also, as correlation is CPU-bound, a more capable VM hardware configuration will have a significant impact. Memory usage was stable at 810MB of RAM, approximately, with the correlator using an average 104MB RAM, with a maximum of 141MB - the rest was used by the RabbitMQ broker.

VII. Conclusion

This paper presented the architecture of the CockpitCI domain-specific IDS for SCADA ICS. Its design departs from the state-of-the-art in CI security, incorporating distributed multi-level correlation, together with domain-specific anomaly detection elements, leveraging the benefits of both signature-based and rogue threat detection. Also, the PIDS brings together a set of heterogeneous detection and analysis components within a common framework with normalized eventing, coordination and orchestration capabilities.

Future developments of the PIDS are focused on the evolution of the Honeypot and Shadow Security Units, as well as support for a more diversified set of SCADA technologies and protocols. Research into complementary analysis techniques is also undergoing, in order to improve detection capabilities.

Acknowledgment

This work was funded by the CockpitCI European FP7 project (FP7-SEC-2011-1 Project 285647).

References

[1] T. Strasser, F. Andrén, J. Kathan, C. Cecati, C. Buccella, P. Siano, P. Leitao, G. Zhabelova, V. Vyatkin *et al.*, "A review of architectures and concepts for intelligence in future electric energy systems," *Industrial Electronics, IEEE Transactions on*, vol. 62, no. 4, pp. 2424–2438, 2015.

[2] C.-S. Cho, W.-H. Chung, and S.-Y. Kuo, "Cyberphysical security and dependability analysis of digital control systems in nuclear power plants," *Systems, Man, and Cybernetics: Systems, IEEE Transactions on*, vol. PP, no. 99, pp. 1–1, 2015.

[3] R. Mitchell and I.-R. Chen, "A survey of intrusion detection techniques for cyber-physical systems," *ACM Comput. Surv.*, vol. 46, no. 4, pp. 55:1–55:29, Mar. 2014.

[4] A. Nicholson, S. Webber, S. Dyer, T. Patel, and H. Janicke, "SCADA security in the light of cyber-warfare," *Computers & Security*, vol. 31, no. 4, pp. 418–436, 2012.

[5] D. Kushner, "The real story of stuxnet," *Spectrum, IEEE*, vol. 50, no. 3, pp. 48–53, March 2013.

[6] ISA, ANSI, "ISA–99.00. 01–2007 security for industrial automation and control systems part 1: Terminology, concepts, and models," *International Society for Automation*, 2007.

[7] B. Zhu, A. Joseph, and S. Sastry, "A taxonomy of cyber attacks on SCADA systems," in *Proc. of the 4th International. Conference. on Cyber, Physical and Social Computing,*, 2011, pp. 380–388.

[8] D. Kang, J. Lee, B. Kim, and D. Hur, "Proposal strategies of key management for data encryption in SCADA network of electric power systems," *International Journal of Electrical Power & Energy Systems*, vol. 33, no. 9, pp. 1521–1526, 2011.

[9] A. Rezai, P. Keshavarzi, and Z. Moravej, "Secure SCADA communication by using a modified key management scheme," *ISA transactions*, vol. 52, no. 4, pp. 517–524, 2013.

[10] G. Dan, H. Sandberg, M. Ekstedt, and G. Bjorkman, "Challenges in power system information security," *IEEE Security & Privacy*, vol. 10, no. 4, pp. 62–70, 2012.

[11] K. Stouffer, S. Lightman, V. Pillitter, M. Abrams, and A. Hahn, "Guide to industrial control system (ICS) security, revision 2 (nist sp 800-82)," NIST, Tech. Rep., 2015.

[12] North American Electric Reliability Council. (2013) Cyber security - standard CIP-002 through 009. [Online]. Available: http://www.nerc.com

[13] United States Nuclear Regulatory Commission. (2009) 73.54 protection of digital computer and communication systems and networks. [Online]. Available: http://www.nrc.gov/reading-rm/doccollections/cfr/part073/part073-0054.html

[14] S. Han, M. Xie, H.-H. Chen, and Y. Ling, "Intrusion detection in cyber-physical systems: Techniques and challenges," *Systems Journal, IEEE*, vol. 8, no. 4, pp. 1049–1059, 2014.

[15] R. Karri and J. Rajendran, "Trustworthy hardware: Identifying and classifying hardware trojans," *Computer*, vol. 43, no. 10, pp. 39–46, 2010.

[16] W. Gao, T. Morris, B. Reaves, and D. Richey, "On SCADA control system command and response injection and intrusion detection," in *Proc. of eCrime Researchers Summit*, 2010, pp. 1–9.

[17] Y. Yang, K. McLaughlin, S. Sezer, T. Littler, E. G. Im, B. Pranggono, and H. Wang, "Multiattribute SCADA-specific intrusion detection system for power networks," *Power Delivery, IEEE Transactions on*, vol. 29, no. 3, pp. 1092–1102, 2014.

[18] L. Maglaras, J. Jiang, and T. Cruz, "Integrated OCSVM mechanism for intrusion detection in SCADA systems," *Electronics Letters*, vol. 50, no. 25, pp. 1935–1936, 2014.

[19] M. Winn, M. Rice, S. Dunlap, J. Lopez, and B. Mullins, "Constructing cost-effective and targetable industrial control system honeypots for production networks," *International Journal of Critical Infrastructure Protection*, 2015.

[20] S. Yin, X. Zhu, and C. Jing, "Fault detection based on a robust one class support vector machine," *Neurocomputing*, vol. 145, pp. 263–268, 2014.

[21] X. Lu, B. Fan, and M. Huang, "A novel LS-SVM modeling method for a hydraulic press forging process with multiple localized solutions," *Industrial Informatics, IEEE Transactions on*, vol. 11, no. 3, pp. 663–670, 2015.

[22] P. Simões, T. Cruz, J. Proença, and E. Monteiro, "Specialized honeypots for SCADA systems," in *Cyber Security: Analytics, Technology and Automation*, ser. Intelligent Systems, Control and Automation: Science and Engineering, M. Lehto and P. Neittaanmäki, Eds. Springer International Publishing, 2015, vol. 78, pp. 251–269.

[23] T. Cruz, J. Barrigas, J. Proenca, A. Graziano, S. Panzieri, L. Lev, and P. Simoes, "Improving network security monitoring for industrial control systems," in *2015 IFIP/IEEE International Symposium on Integrated Network Management*, May 2015, pp. 878–881.

[24] L. A. Maglaras, J. Jiang, and T. Cruz, "Combining ensemble methods and social network metrics for improving accuracy of {OCSVM} on intrusion detection in {SCADA} systems," *Journal of Information Security and Applications*, 2016.

[25] Dept. of Homeland Security ICS-CERT. (2012) Schneider electric modicon quantum vulnerabilities (update b). [Online]. Available: https://ics-cert.us-cert.gov/alerts/ICS-ALERT-12-020-03B

JOURNAL OF INFORMATION SECURITY AND APPLICATIONS 30 (2016) 15–26

Available online at www.sciencedirect.com

ScienceDirect

journal homepage: www.elsevier.com/locate/jisa

Combining ensemble methods and social network metrics for improving accuracy of OCSVM on intrusion detection in SCADA systems

Leandros A. Maglaras [a,*], *Jianmin Jiang* [b], *Tiago J. Cruz* [c]

[a] *School of Computer Science and Informatics, De Montfort University, Leicester, UK*
[b] *Department of Computing, University of Surrey, Guildford, Surrey, UK*
[c] *Department of Informatics Engineering, University of Coimbra, Portugal*

ARTICLE INFO

Article history:
Available online 4 May 2016

Keywords:
OCSVM
Intrusion detection
SCADA systems
Social analysis

ABSTRACT

Modern Supervisory Control and Data Acquisition (SCADA) systems used by the electric utility industry to monitor and control electric power generation, transmission and distribution are recognized today as critical components of the electric power delivery infrastructure. SCADA systems are large, complex and incorporate increasing numbers of widely distributed components. The presence of a real time intrusion detection mechanism, which can cope with different types of attacks, is of great importance in order to defend a system against cyber attacks. This defense mechanism must be distributed, cheap and above all accurate, since false positive alarms or mistakes regarding the origin of the intrusion mean severe costs for the system. Recently an integrated detection mechanism, namely *IT-OCSVM*, was proposed, which is distributed in a SCADA network as a part of a distributed intrusion detection system (DIDS), providing accurate data about the origin and the time of an intrusion. In this paper we also analyze the architecture of the integrated detection mechanism and we perform extensive simulations based on real cyber attacks in a small SCADA testbed in order to evaluate the performance of the proposed mechanism.

1. Introduction

Cyber-physical systems are becoming vital for modernizing national critical infrastructure systems. Cyber attacks often target valuable infrastructure assets, taking advantage of architectural/technical vulnerabilities or even weaknesses in defense systems. Most weaknesses in critical infrastructures (CIs) arise from the fact that many adopt off-the-shelf technologies from the IT world, without a significant change in terms of the operator mindset, thus remaining based on the "airgap" security principle that falsely assumes that an apparently isolated and obscure system is implicitly secure. Consequently, once a system is open to receiving off-the-shelf solutions, this increases its exposure to cyber-attacks.

The proliferation of new technologies, especially Internet-like communication networks, may introduce some new threats to the security of a smart grid. In such a network there are three crucial aspects of security that may be threatened due to the CIA-triad, these being confidentiality, integrity, and availability (Bennett and Highfill, 2008). Confidentiality is the property that information is not made available or disclosed to

* *Corresponding author.* School of Computer Science and Informatics, De Montfort University, Leicester, UK. Tel.: +44 116 2078483.
E-mail address: leandros.maglaras@dmu.ac.uk (L.A. Maglaras).
http://dx.doi.org/10.1016/j.jisa.2016.04.002
2214-2126/

unauthorized individuals, entities or processes. An attack on this occurs when an unauthorized person, entity or process enters the system and accesses the information. Integrity refers to safeguarding the accuracy and completeness of assets, which ensures that the information in the system will not be modified by attacks. Availability pertains to the property of being accessible and usable upon demand by an authorized entity. The resources need to be kept accessible at all times to authorized entities or processes.

Beyond cyber threats like malware, spyware, and computer viruses that currently threaten the security of computer communication networks, the introduction of new and distributed technologies, such as smart meters, sensors, and other sub-networks, can bring new vulnerabilities to a smart grid (McDaniel and McLaughlin, 2009). In the three main control systems of a CI, the SCADA is the central nerve system that constantly gathers the latest status from remote units. The communication system for wide-area protection and control of a power grid can be blocked or cut off due to component failures or communication delays. If one of the crucial communication channels fails to connect in the operational environment, inability to control or operate important facilities may occur with the possibility of power outages. In this situation, the effect of some widely known attacks can have devastating consequences on SCADA systems. Moreover, the design of the SCADA systems is different from conventional IT networks, even when based on the same physical technology, such as Ethernet networks. That is, industrial control-specific protocols are used in SCADA systems, where a limited number packet types are exchanged between entities of the network.

Intrusion detection systems (IDS) can be classified into centralized intrusion detection systems (CIDS) and distributed intrusion detection systems (DIDS) by the way in which their components are distributed (Balasubramaniyan et al., 1998). A CIDS is such that the analysis of the data will be performed in some fixed locations without considering the number of hosts being monitored (Kumar, 1995), while a DIDS is composed of several IDS over large networks whose data analysis is performed in a number of locations proportional to the number of hosts. As one part of an intrusion detection system, the DIDS has specific advantages over the CIDS. For instance, it is highly scalable and can provide gradual degradation of service, easy extensibility and scalability (Crosbie and Spafford, 1995). It is evident that the development of distributed IDS specifically designed for SCADA systems, being able to ensure an adequate balance between high accuracy, low false alarm rate and reduced network traffic overhead, is needed. The above discussion clearly indicates that specific intrusion detection systems that reassure both high accuracy, low rate of false alarms and decreased overhead on the network traffic need to be designed for SCADA systems.

1.1. *Motivation*

Among other approaches, neural networks, support vector machines, K-nearest neighbor (KNN) and the hidden Markov model can be used for intrusion detection. While existing signature-based network IDS, such as Snort or Suricata, can be effective in SCADA environments, they require specific customization for such a purpose. Also, they are not effective against rogue threats for which known patterns or signatures are not known. OCSVM principles have shown great potential in the area of anomaly detection (Maglaras and Jiang, 2014; Wang et al., 2004), being a natural extension of the support vector algorithm in the case of unlabeled data, especially for the detection of outliers.

Social network analysis (SNA) can be used in order to discover security policy breaches in a network and refers to the use of network theory to analyze social networks. That is, it views social relationships in terms of network theory, consisting of nodes, representing individual actors within the network, and ties which represent relationships between the individuals, such as friendships, kinships, organizations and sexual relationships. By using comparative metrics of network's structure during normal and abnormal operation, we can discover security policy breaches in a network. One can assume that network communication between nodes constitutes a social network of users and their applications, so the appropriate methods of social network formal analysis can be applied (Kołaczek and Prusiewicz, 2011). In on-line social systems perpetrators of malicious behavior often display patterns of interaction that are quite different from regular users, which can be identified through the application of anomaly detection techniques. Thus, in accordance with García-Teodoro et al. (2009) and Gogoi et al. (2011), network anomalies can be defined as patterns of interaction that significantly differ from the norm and in order to capture the appropriate patterns of interaction, specific aspects of entities' behavior are used (e.g. email analysis). Similar to this, in a SCADA system, individual entities demonstrate a quite different communication behavior when infected, in terms of mean packet generation frequency (traffic burst), protocol distribution or interaction pattern.

Discovering anomalies in the context of a network system is a challenging issue due to the complexity of the environment and the different nature of the induced attacks. Regarding node behavior related decisions, it makes sense to ask more than one decision mechanism since this practice assures a more trusted final decision. Ensemble systems of classifiers are widely used for intrusion detection in networks (Curiac and Volosencu, 2012; Shieh and Kamm, 2009). These aim to include mutually complementary individual classifiers, which are characterized by high diversity in terms of classifier structure (Tsoumakas et al., 2004), internal parameters (Kim and Kang, 2010) or classifier inputs (Krawczyk and Woźniak, 2014).

In real time systems, in addition to fast response and accuracy, limited communication between detection modules is also desirable. By sending an explicit message for every anomaly detected, the intrusion detection mechanism will flood the medium with messages that will cause a delay in the communication between entities in the SCADA system. Moreover, since the detection mechanism needs to be sited at several locations in the SCADA system in order to recognize the intrusion near the origin, the communication overhead caused by the detection mechanism is further increased. One solution is the addition of a control channel, whereby these messages can be communicated without affecting the system's performance, but this is not always feasible. For this reason, an aggregation mechanism that groups initial alerts and sends a limited

number of messages reporting the fault/intrusion accurately and on time is needed.

1.2. *Contributions*

The present article analyzes and evaluates the performance of a recently proposed intrusion detection mechanism, namely the *IT-OCSVM* (Maglaras et al., 2014), against the baseline OCSVM method. The mechanism uses a central OCSVM and a cluster of automatically produced ones, one for each source that induces significant traffic in the system, an embedded ensemble mechanism, social metrics, an aggregation method and a k-means clustering procedure that categorizes aggregated alerts. The mechanism runs in a distributed way and produces dedicated IDMEF (Intrusion Detection Message Exchange Format) messages that report the severity of the attack detected. The proposed mechanism is incorporated in a distributed IDS (intrusion detection system) communicating with other detection and management components of the system. *IT-OCSVM* is evaluated using datasets extracted from a testbed that mimics a small scale SCADA system under normal and malicious operation. The evaluation of the proposed method attests the superiority of the new structure in terms of accuracy, false alarm rate and system overhead. The article makes the following contributions:

- Analyzes the architecture of the IT-OCSVM mechanism that was initially presented in Maglaras et al. (2014).
- Describes in depth the different components that comprise the IT-OCSVM mechanism.
- Describes how distributed agents that run the IT-OCSVM mechanism be integrated in a Perimeter Intrusion Detection System (PIDS).
- Evaluates the performance of the IT-OCSVM mechanism through extensive simulations using datasets that are extracted from a testbed that mimics a small-scale SCADA system in terms of accuracy and communication overhead.
- Evaluates the computational cost and time overhead of the IT-OCSVM mechanism.

The rest of this article is organized as follows: Section 2 presents the use of OCSVM in SCADA systems; Section 3 describes the IT-OCSVM mechanism; Section 4 presents the simulation environment and results; and Section 5 concludes the article.

2. OCSVM for intrusion detection in SCADA systems

SCADA systems (Fig. 1) that tie together decentralized facilities, such as power, oil and gas pipelines, as well as water distribution and wastewater collection systems, were designed to be open, robust, and easily operated and repaired, but not necessarily secure. Cyber-attacks against these systems are considered extremely dangerous for CI operation and must be addressed in a specific way (Zhu et al., 2011).

The one-class classification problem is a special case of the conventional two-class classification problem, where only data from one specific class are available and well represented. This

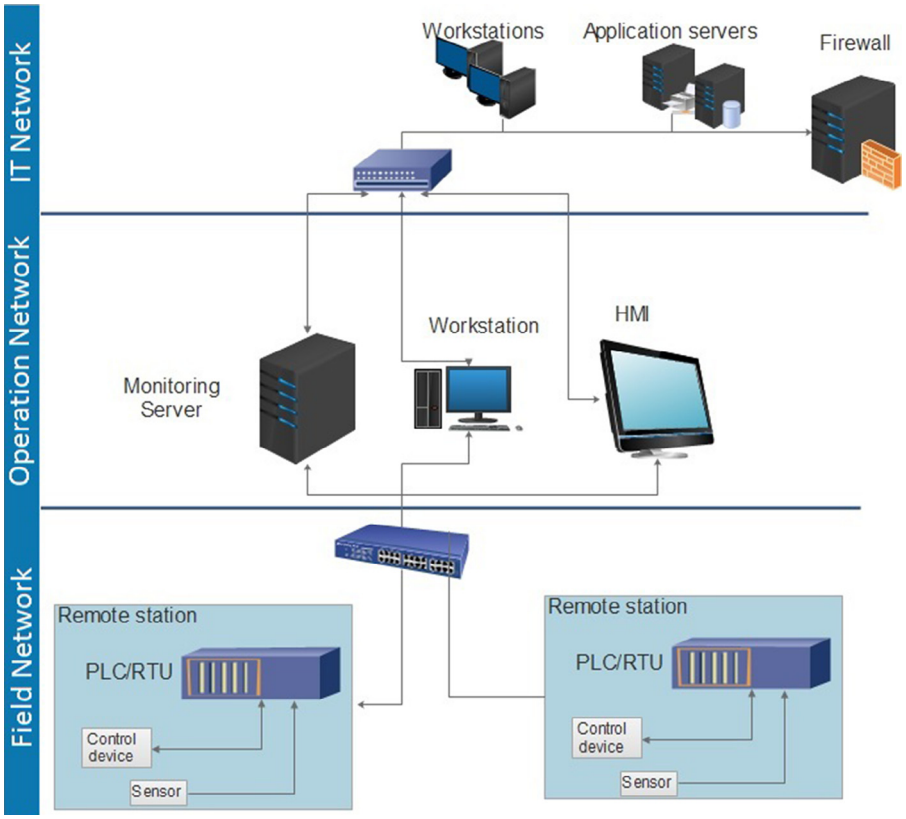

Fig. 1 – A typical SCADA system.

class is called the target class. Another class, which is called the outlier class, can be sampled very sparsely, or not even at all. This smaller class contains data that appear when the operation of the system varies from normal due to a possible attack. The one class Support Vector Machine (OCSVM) (Jiang and Yasakethu, 2013) possesses several advantages for processing SCADA environment data and automates SCADA performance monitoring, which can be highlighted as:

- In the case of SCADA performance monitoring, which patterns in the data are normal or abnormal may not be obvious to operators. Since OCSVM does not require any signatures of data to build the detection model, it is well suited for intrusion detection in SCADA environments.
- Since the detection mechanism does not require any prior information of the expected attack types, OCSVM is capable of detecting both known and unknown (novel) attacks.
- In practice, training data taken from SCADA environment could include noise Samples and most of the classification based intrusion detection methods are very sensitive to noise. However, the OCSVM detection approach is robust to noise samples in the training process.
- Algorithm configuration can be controlled by the user to regulate the percentage of anomalies expected.
- Due to the low computation time, OCSVM detectors can operate fast enough for online SCADA performance monitoring.
- Typical monitoring of data of SCADA systems consists of several attributes and OCSVM is capable of handling multiple attributed data.

3. IT-OCSVM detection mechanism

The main purpose of the *IT-OCSVM* detection mechanism is to perform anomaly detection in a time-efficient way, with good accuracy and low overhead, within a temporal window adequate for the nature of SCADA systems. In order to achieve the aforementioned goals several operation stages need to be carried out.

- **Pre-processing** of raw input data in order to feed the *IT-OCSVM* module. The attributes in the raw data of the testbed contain all forms: continuous, discrete, and symbolic, with significantly varying resolution and ranges. Most pattern classification methods are not able to process data in such a format. Hence, pre-processing is required before pattern classification models can be built. This consists of two steps: the first involves mapping symbolic-valued attributes to numeric-valued attributes and the second is implemented scaling. Different pre-processing techniques are used based on the characteristics of each feature type (Lazarevic et al., 2003).
- **Selection of the most appropriate features** for the training of the *IT-OCSVM* model. These features are divided into content and time based features. Since the majority of DoS and probing attacks may use hundreds of packets, time-based features are mostly used.
- **Creation of cluster of OCSVM models** that are trained on discrete sources. There are many slow probing attacks that scan the hosts using much larger intervals, thus being able to merge into the overall traffic in the network. As a consequence, these attacks cannot be detected using derived time based features and in order to capture them, the raw data after arriving in the module are split into different datasets according to the sender of the packet. An OCSM module is created and trained for each split dataset. The cluster of split OCSVMs runs in parallel with the central OCSVM and produces errors targeted to the specific source.

It is important to mention that a split OCSVM is not created for each source, but only for those sources that produce high traffic in the network during the training period. In order to separate significant nodes, a threshold $P_{packets}$ is used and every source that produces a number of packets over this threshold during the training period is marked as a significant node. If during the testing period a source is showing big activity, while not being marked as significant, a medium alarm is fired for it.

- **Testing of the traffic dataset** that contains malicious attacks. Based on the models created from the training phase, the new dataset is tested against normal patterns. Each OCSVM module returns a function f that takes the value +1 in a region capturing normal data points (i.e. for events implying normal behavior of the SCADA system) and takes a negative value elsewhere (i.e. for events implying abnormal behavior of the SCADA system).
- **Ensemble of classifiers.** The initial outcomes of the different OCSVM modules are combined by the ensemble based mechanism that uses mean majority voting.
- **Social analysis.** Social network analysis is executed based on the network traces and Spearman's rank correlation coefficient is used in order to add weight to alerts produced from different sources.
- **Fusion of the information.** Due to the possible existence of multiple anomalies in a SCADA system, the outcomes of the different models are gathered and classified in terms of importance. This importance is derived through aggregation and k-means clustering of the outputs.
- **Communication of the mechanism**. In order to cooperate with other components of the IDS, the mechanism sends IDMEF files. The created files describe the nature of the alert in terms of importance, the position in the system, time, etc.

3.1. *Attribute extraction*

Feature extraction is essential in a classification problem. Attributes in the network trace datasets have all forms: continuous, discrete, and symbolic, with significantly varying resolution and ranges. Most pattern classification methods are not able to process data in such a format. Hence pre-processing is required before pattern classification models could be built. Pre-processing consists of two steps: The first step involved mapping symbolic-valued attributes to numeric valued attributes and the second step implemented scaling. The main advantage of scaling is to avoid attributes in greater numeric ranges dominating those in smaller numeric ranges. Another advantage is to avoid numerical difficulties during the

Table 1 – Features of the central OCSVM model.

Network data feature	Type of feature
Packet size	Content based
Rate	Time based
$No_{packets-dst}$	Time based
$No_{packets-src-dst}$	Time based
$No_{ARP-packets}$	Time based

calculation. Based on a previous analysis of data we selected some initial features that are used as attributes for our OCSVM model.

3.1.1. *Central OCSVM*

The features and their corresponding network data features are shown in Table 1. Some features of the data are derived features, which are useful in distinguishing normal connection from attacks. These features are either nominal or numeric. The smooth packet growth over time represents a normal behavior of the system when no malicious data are detected and the slope of the line can vary from time to time, depending on the load of the system. When malicious data are broadcasted over the network, the accumulative packet size may rise, but since in many situations intruders send 0 length packet sizes (ack/nack packets), this feature by itself is not enough to detect these situations. A burst in traffic injected in the system is another characteristic of an intrusion. Infected nodes may broadcast messages flooding the system with messages that are of no use and block the normal operation of the network. Thus, the first two attributes that we used for this initial training/testing phase were rate and packet size. The values were scaled to the range [0, 1].

The rate (1st attribute) was calculated using Eq. (1):

$$Rate_{scaled} = \frac{Time\ difference}{Max\ Time\ Difference} \quad (1)$$

Time difference is calculated by the difference of time of current packet and the time of previous packet injected in the system.

The packet size (2nd attribute) was scaled using Eq. (2):

$$Packetscaled = \frac{Packet\ Size}{Max\ Packet\ Size} \quad (2)$$

Moreover, since majority of DoS and probing attacks may use hundreds of packets, we have constructed time-based features that attempt to capture the previous recent packets with similar characteristics. Feature 3 examines only the packets in the last 10 packets that have the same destination host as the current packet (same host feature). Feature 4 examines only the packets in the past 10 packets that have both same source and destination as the current packet. For producing feature 5, each row of the dataset is assigned a value indicating the number of ARP packets in the last 10 packets.

3.1.2. *Cluster of split OCSVMs*

There are many slow probing attacks that scan the hosts using much larger interval and merge in the overall traffic in the network. As a consequence, these attacks cannot be detected using derived time based features. In order to capture these types of attacks, we modified the OCSVM module to automatically create a cluster of split OCSVMs (see Fig. 2). The features of the split OCSVM modules are a subset of the central OCSVM and include features 1, 2, and 5. Moreover, the split datasets, each consisting of the packets send only by an individual source, add to the detection mechanism the diversity that is crucial in order to improve the overall eficiency (dataset splitting; Krawczyk and Woźniak, 2014).

3.2. *Ensemble system*

Ensemble methods can be differentiated according to the extent each classifier affects the others. This property indicates whether the classifiers are dependent or independent. The first situation occurs when the outcomes of a certain classifier affect the creation of the next (Menahem et al., 2013) and the latter, when each classifier is built independently and their results are combined in some fashion (Tax and Duin, 2001). Our proposed mechanism uses an independent ensemble mechanism. Apart from the central OCSVM, which is trained on the entire dataset, a cluster of split OCSVMs is automatically created through the decomposition of the traffic dataset into disjoint subsets 2. The idea is based on the work of Chawla et al. (2004), who achieved high accuracy by building thousands of classifiers trained from small subsets of data in a distributed environment and through this decomposition, high diversity of the methods is achieved. Diversity is an essential feature of an ensemble mechanism in order to achieve high accuracy (Tumer and Ghosh, 1996).

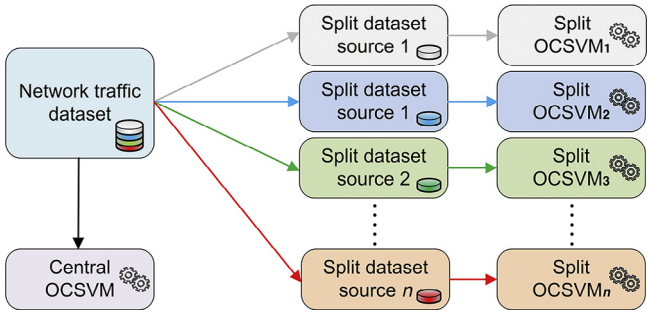

Fig. 2 – Decomposition of the traffic dataset to disjoint subsets.

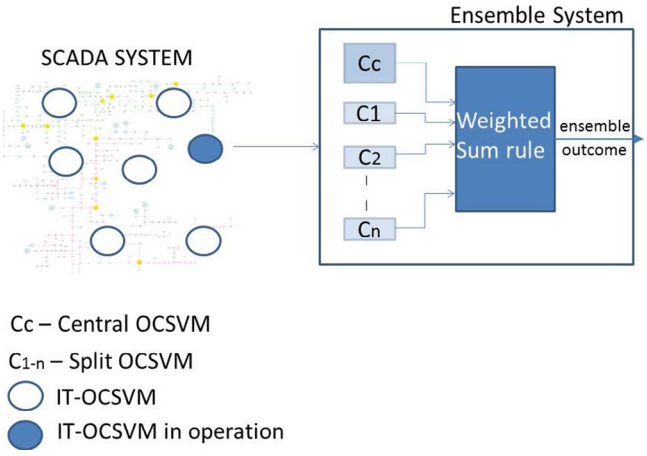

Fig. 3 – Ensemble based system.

A combination procedure is then applied so as to produce a single classification for a given instance. There exist many ensemble classification methods, e.g. majority voting, performance weighting, distribution summation and order statistics. These methods have some advantages and disadvantages in terms of accuracy and computational cost. In a real time system, both parameters are crucial and for this reason a simple but effective ensemble mechanism must be chosen (Rokach, 2010). In order to achieve a balance between these two parameters, the outcomes of both the central and the split OCSVMs are combined using a simple algebraic weighted sum rule using Eq. (3). The ensemble based mechanism is represented in Fig. 3.

$$q_e(i, j) = \sum_{n=1}^{N} w_i d_t(i, j) \tag{3}$$

where $d_t(i, j)$ is the outcome of each individual classifier n for sample data i that originates from node j and w_i is the weight given to each classifier.

The values that are produced from this stage are weighted using the social analysis module, according to the Spearman's rank coefficient for every significant source. These values are then fed into the aggregation module.

3.3. *Social analysis module*

Deviation from the normal protocol operation in communication networks has received considerable attention from the research community in recent years. A malicious node may use the vulnerabilities of the system architecture to perform different kinds of attacks. Hence, for network reliability it is necessary to develop an efficient technique to detect misbehaving clients in a timely manner and the correlation coefficients between entities can effectively detect malicious nodes (Hamid et al., 2008).

In order to enhance the performance of the OCSVM module, parallel statistical algorithms are executed. For each significant source that is detected during training, a list of the protocols used is created, which are ranked and stored along with the IP/MAC addresses of each source. During the testing phase the same procedure is executed, thus producing a list

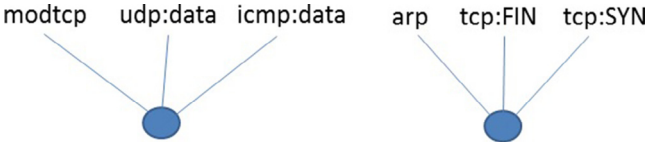

Fig. 4 – Most used protocols used by a node during normal (left) and abnormal (right) operation.

of the protocols used for each significant source. The two lists are compared using Spearman's correlation coefficient (Spearman, 1904).

$$p = 1 - \frac{6\sum d_i^2}{n(n^2 - 1)} \tag{4}$$

The final output is a number q_j for each source j that indicates whether there is a differentiation in the way that each source behaves during the training and testing period. The output of the method is a value between 1 and 0, with a number close to 1 indicating similar behavior of the source both in the training and testing sessions. This value is used in order to give additional significance to alerts produced from the specific source from the OCSVM modules (see Eq. 5). In Fig. 4 an example of the protocols that are mainly used during normal and abnormal operation of a node is presented.

$$q_s(i, j) = \frac{q_e(i, j)}{p_j}, \forall\ q_e(i, j)\ \textit{with source node}\ j \tag{5}$$

3.4. *Fusion of alarms — final outputs*

The OCSVM module produces one initial alarm for any deviation in network traffic from normal, based on the training models that it has created. In real time systems in addition to fast response and accuracy, limited communication between the detection modules is desirable. For this reason we implement a fusion procedure which groups alerts per source node and gives final scores to aggregated alerts based on the initial values and the number of similar initial alerts.

The first stage of fusion consists of an aggregation mechanism that groups individual alerts according to their origin. Each alert i has an initial value q_i (and a source j) based on the procedures described in Sections 3.2 and 3.3. Using Eq. (6) each aggregated alert j is assigned two values qa_j and qb_j that represent its severity. This severity comes from both the sum of the values of the initial alerts and the number of attacks that originate from the same node.

$$qa_j = \sum_i q_s(i, j), \quad qb_j = \sum_i 1, \forall\ q_s(i, j)\ \textit{with source node}\ j \tag{6}$$

During the second stage of fusion the system uses K-means clustering so as to divide the alarms into possible, medium and severe. The k-means clustering algorithm is one of the simplest and most commonly used clustering algorithms. It is a partitional algorithm that heuristically attempts to minimize the sum of squared errors.

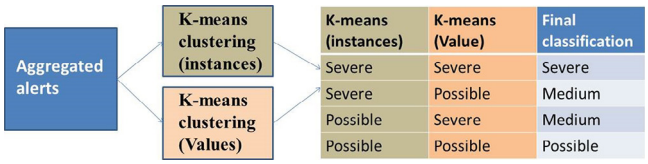

Fig. 5 – K-means clustering module.

$$SSE = \sum_{k=1}^{K} \sum_{j=1}^{N_k} \|qa_j - \mu_k\|^2 \quad (7)$$

where N_k is the number of instances belonging to cluster k and μ_k is the mean of cluster k, calculated as the mean of all the instances belonging to cluster i,

$$\mu_{k,i} = \frac{1}{N_k} \sum_{q=1}^{N_k} qa_q,\ j\ \forall j \quad (8)$$

The algorithm begins with an initial set of cluster centers and in each iteration each instance is assigned to its nearest cluster center according to the Euclidean distance between the two. We use two k-means algorithms that run in parallel which partition alerts into two categories: possible and severe. These decisions are then combined according to the above equations and are assigned their final classification: possible, medium or severe, according to Fig. 5.

Using the fusion procedure, the final alarms that the system produces are significantly decreased and all the sources with suspect behavior are reported. The whole procedure is described in Fig. 6.

In order to cooperate with the other modules, the *IT-OCSVM* mechanism is integrated in the *PID* system and communicates with the other modules created under the "Cockpit CI project" using IDMEF (Debar et al., 2007) files. The IDMEF defines the experimental standard for exchanging intrusion detection related events and a typical IDMEF file produced by our system is shown in Fig. 7. The IDMEF message contains information about the source of the intrusion, the time of the intrusion detection, the module that detected the problem and a classification of the detected attack. Knowledge of the source node where the intrusion originates is a very important feature an IDS system must have. Once the infected node is spotted, the infection can be limited by the isolation of this node from the rest of the network. Fast and accurate detection of the source node of a contamination is crucial for the correct function of an IDS.

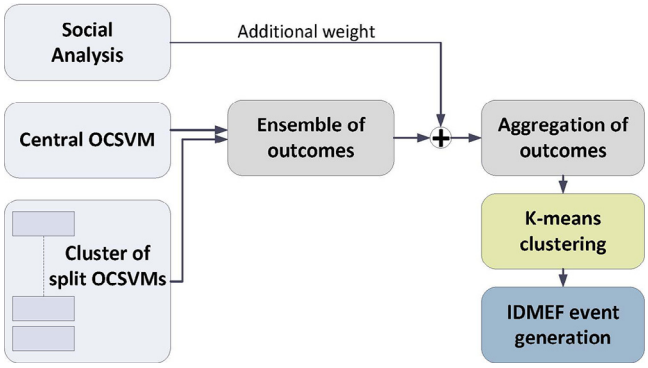

Fig. 6 – Architecture of detection mechanism.

3.5. *Nature of the trial*

The trial is conducted offline with the use of two datasets extracted from the testbed (Fig. 8). The testbed architecture mimics a small-scale SCADA system, comprising the operations and field networks, which include a Human–Machine Interface Station (for process monitoring), a managed switch (with port monitoring capabilities for network traffic capture), and two Programmable Logic Controller Units, for process control. The *NIDS* and *IT-OCSVM* modules are co-located on the same host, being able to intercept all the traffic flowing in the network.

Three kinds of attacks are evaluated:

- **Network scan attack** — In a typical network scan attack, the attacker uses TCP/FIN scan to determine if ports are

```
<?xml version="1.0" encoding="UTF-8"?>
- <idmef:IDMEF-Message version="1.0" xmlns:idmef="http://iana.org/idmef">
  - <idmef:Alert>
    - <idmef:Analyzer analyzerid="test">
      - <idmef:Node category="unknown">
          <idmef:location>IT Network</idmef:location>
          <idmef:name>OCSVM</idmef:name>
        </idmef:Node>
      </idmef:Analyzer>
      <idmef:CreateTime ntpstamp="0x1130fdd3.0xa0000000">2014-08-19T16:19:43+01:00</idmef:CreateTime>
    - <idmef:Source>
      - <idmef:Node>
        - <idmef:Address category="ipv4-addr">
            <idmef:address>172.27.224.32</idmef:address>
          </idmef:Address>
        </idmef:Node>
      </idmef:Source>
    - <idmef:Target>
      - <idmef:Node>
        - <idmef:Address category="ipv4-addr">
            <idmef:address>172.27.224.3</idmef:address>
          </idmef:Address>
        </idmef:Node>
      </idmef:Target>
      <idmef:Classification text="SEVERE ALARM"/>
    </idmef:Alert>
</idmef:IDMEF-Message>
```

Fig. 7 – Typical IDMEF message produced by the IT-OCSVM mechanism.

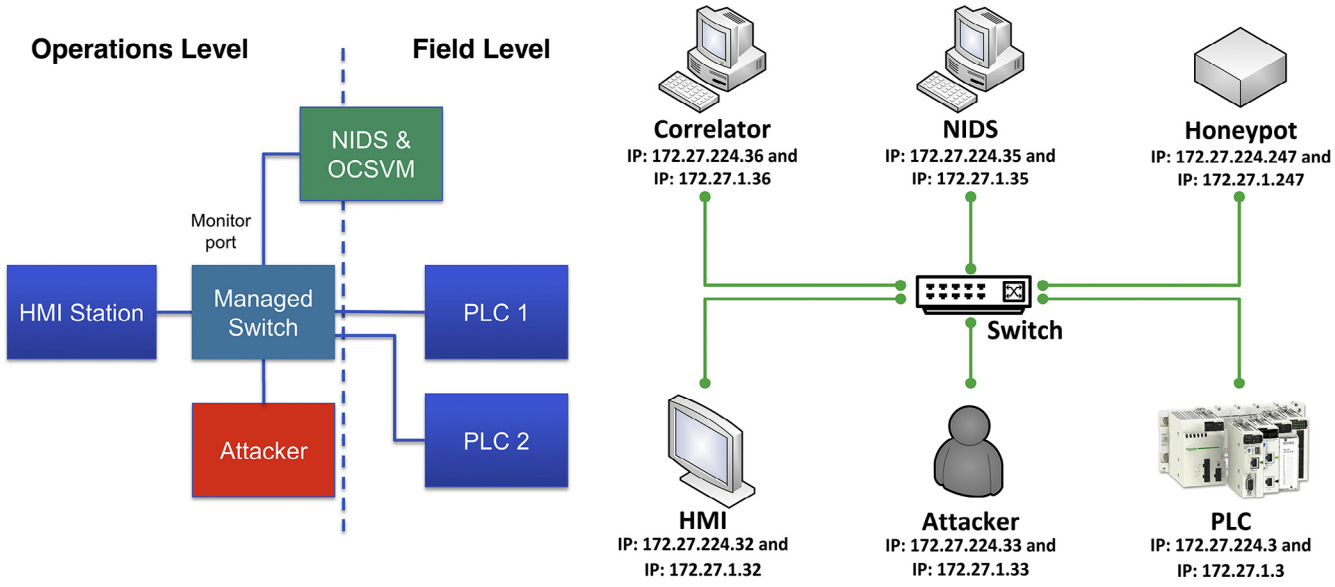

Fig. 8 – Architecture of the testbed.

closed to the target machine; closed ports answer with RST packets while open ports discard the FIN message. FIN packets blend with background noise on a link and are hard to detect.

- **ARP cache spoofing — MITM attack ARP cache spoofing** is a technique where an attacker sends fake ARP messages. The aim is to associate the attacker's MAC address with the IP address of another host, thus causing any traffic meant for that IP to be sent to the attacker instead. The attacker could choose to inspect the packets, modify data before forwarding (**man-in-the-middle attack**) or launch a denial of service attack by causing some of the packets to be dropped.
- **DoS attack** — Network flood refers to the situation where the attacker floods the connection with the PLC by sending SYN packets.

4. Results and analysis

Each OCSVM that is trained produces a separate model file which is used in order to classify new data as normal or malicious. All OCSVMs use Gaussian RBF Kernel functions with default parameters $\sigma = 0.01$ and $\nu = 0.001$. The parameters used during evaluation of the proposed IT-OCSVM detection method are shown in Table 2.

During execution of the proposed detection mechanism, a separate file that contains information about the split sources is also created. This file is used in order to split the testing dataset according to the sources that are categorized as important during training. A snapshot of the sources file is shown in Fig. 9. For each row, the IP/MAC address of the source is followed by the 5 most used protocols during the training period, which are sorted in descending order. These protocols are compared with those used by the same source in the testing period in order to calculate the Spearman's correlation coefficient value for each separate source.

Table 2 – Evaluation parameters.

Independent parameter	Default value
ν	0.001
σ	0.01
$P_{packets}$ – (No of split OCSVMs)	$\frac{1}{100}(5)$
Ensemble mechanism	Weighted sum value
k-means	2 stages

4.1. *Initial network traffic analysis*

In Fig. 10 the rate of the injected packets in the system during normal and abnormal operation of the SCADA system is monitored. In the lower part of the figure, we can observe that when the DOS attack takes place, the rate of the packets injected into the system is much higher compared to a normal operation period.

MITM and network scan attacks, on the other hand, do not have such profound consequences on the traffic rate, since they use few messages (MITM) or they merge with the overall traffic. As can be observed from Fig. 11, an ARP spoofing attack can be identified by using feature 5 in the overall dataset. Apart from the actual attack, there are also other instances where the central OCSVM would probably fire an alarm due to high values of this feature.

When subsets are used, the occurrence of this attack is more obvious. In Fig. 12 the values of feature 5 over time for two sources that induce significant traffic in the system are shown. The left diagram presents the number of ARP packets sent by

```
172.27.224.32,eth:ip:tcp:mbtcp:modbus,eth:ip:tcp, , ,
172.27.224.3,eth:ip:tcp:mbtcp:modbus,eth:ip:icmp:data, , ,
172.27.224.33,eth:ip:icmp:data, , , ,
10.3.3.28,eth:ip:udp:data, , , ,
d0:7e:28:8e:40:9b,eth:llc:stp, , , ,
```

Fig. 9 – Sources file created by the OCSVM module.

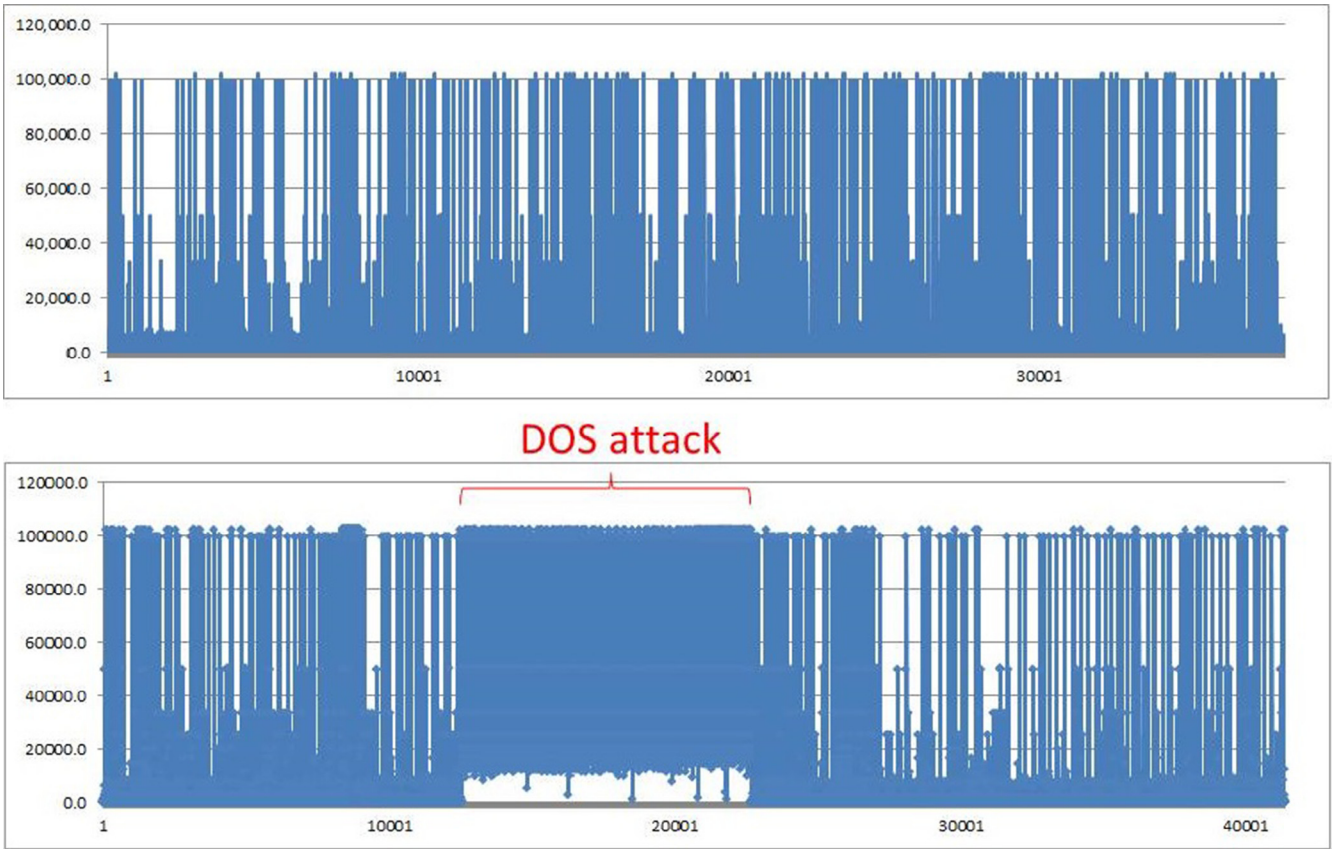

Fig. 10 – Rate of packets during normal and abnormal operation time.

the intruder during the testing period, while the right presents the same distribution for a normal node of the system. The difference in the performance of a node under attack is more evident using the split datasets.

Moreover, the Spearman's rank correlation coefficient of the source that executes a scan attack (Network scan or ARP scan) is heavily affected since node behavior deviates from normal in terms of protocol distribution, thus enhancing the detection capability of our mechanism.

4.2. *Method evaluation*

This subsection describes the performance of the proposed OCSVM based intrusion detection algorithm for the simulated data. During the testing period, several attack scenarios are simulated in the testbed, which include a network scan, a network flood and an MITM attack. Since the attacks are performed during different time periods, we divide the testing dataset into several smaller ones, each containing a different

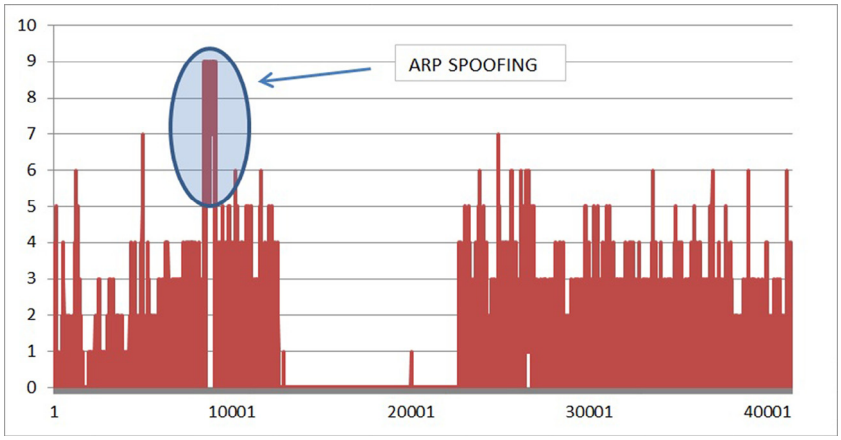

Fig. 11 – ARP spoofing attack — overall dataset.

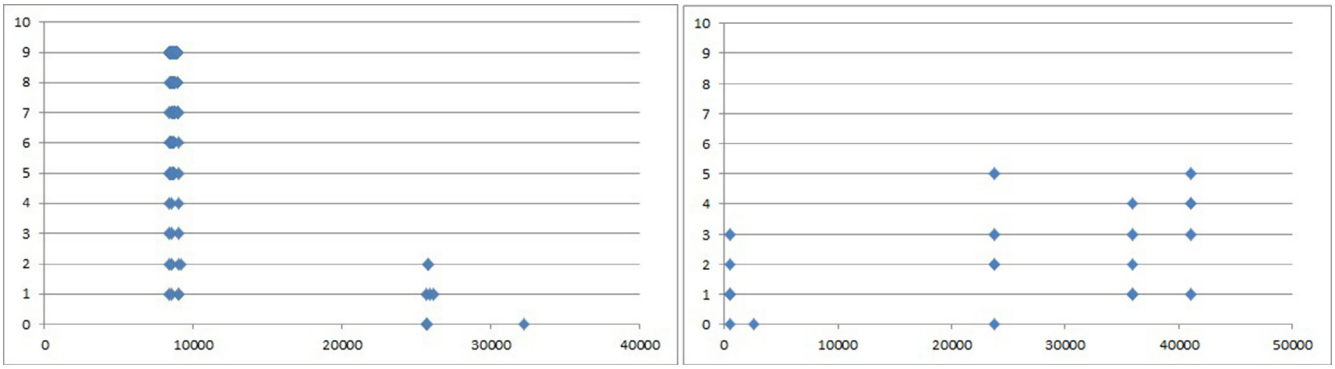

Fig. 12 – ARP spoofing attack — split datasets.

attack. The testing data consist of normal and attack data and the composition of the datasets is as follows:

- Testing set-A′: 1–5000: Normal data
- Testing set-B′: 5000–10,000: Normal data + **Arp spoofing** attack + **Network scan**
- Testing set-C′: 10,000–25,000: Normal data + **Flooding Dos attack** + **Network scan**
- Testing set-D′: 25,000–41,000: Normal data + **MITM attack**

The IT-OCSVM method, as shown in Maglaras et al. (2014), performs well in terms of detection accuracy and false positive rate 3 (Table 3).

4.3. *Ensemble mechanism. Parameter* $P_{packets}$

The ensemble module affects the performance of the IT-OCSVM. Discrete OCVSMs that are created by the mechanism have an impact on the accuracy of the detection mechanism. Regarding which, the cluster of automatically produced OCSVMs can be significantly large (one OCSVM per source) or very small (a total of one or two OCSVMs) according to the value of the parameter $P_{packets}$. Fig. 13 shows the relation between the value of the threshold parameter $P_{packets}$ and the number of created OCSVMs as well as how this affects the performance of the detection mechanism.

The parameter $P_{packets}$ is given a value as a proportion of the number of rows of the entire training dataset. According to Fig. 13, it is evident that an appropriate selection of parameter $P_{packets}$ is essential for the correct operation of the proposed mechanism. When the parameter is given a very big value, then the created OCSVMs are too few to improve accuracy and the performance of the mechanism deteriorates. When this value is extremely big $\frac{1}{10}$, then the mechanism degrades to a simple OCSVM module. For values relatively low, the number of created OCSVMs grows without an improvement in systems accuracy. As observed in Fig. 13, when the number of created OCSVMs is relatively large, the accuracy of the mechanism drops. As split training sets are created for sources with no significant traffic in the training period, the OCSVMs that are created are not correctly trained and instead of helping, they harass the decision procedure. For values of the threshold parameter between $\frac{1}{40}$ and $\frac{1}{100}$, the accuracy of the detection mechanism is optimal. In fact, for these values, the accuracy of the IT-OCSVM mechanism compared to the initial OCSVM is 6% better. That is, the simple OCSVM achieves accuracy of 90.7%, while the proposed IT-OCSVM reaches 96.4% (Fig. 13). Apart from the profound improvement in terms of accuracy, the proposed mechanism has other advantages compared to a simple OCSVM module. It creates a decreased number of alarms and also categorizes these according to their severity, as described in the next subsection.

Table 3 – Performance evaluation of the IT-OCSVM module.

Dataset	DA	FAR
A	98.81%	1.18%
B	94.6%	3.25%
C	95.20%	1.51%
D	96.37%	2.3%
All	96.3%	2.5%

4.4. *Impact of the fusion mechanism*

The fusion of the alarms produced by the individual OCSVMs consists of two stages. The first is aggregation of

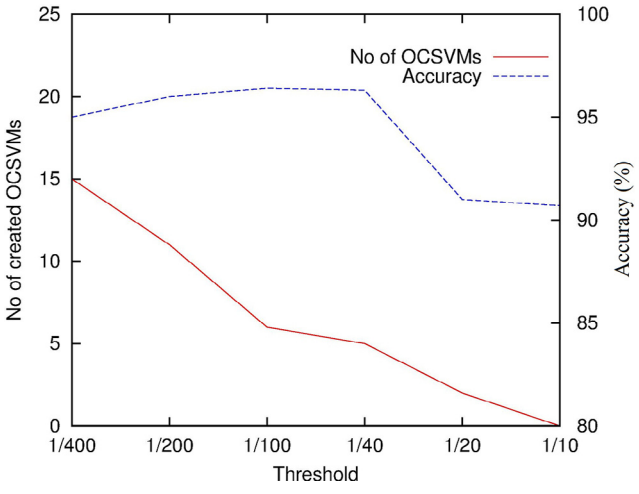

Fig. 13 – Number of automatically created OCSVMs against accuracy of the mechanism.

Table 4 – Aggregated alarms produced by IT-OCSVM are significantly decreased compared to the initial ones.

Dataset	Initial alarms	Aggregated alarms
A	129	16
B	658	21
C	9,273	18
D	203	16
All	10,507	22

alarms per source and the second is the clustering of them using a two-stage k-means clustering algorithm, as described in Section 3.4. The outcome of the fusion procedure is the deduction of the communicated alarms in the system and also their classification as possible, medium and severe. In Table 4 the number of initial and final aggregated alarms is presented. It can be observed that the number of final alarms is significantly lower compared to the initial ones, thus reducing the communication costs that such a distributed mechanism have in the network.

The IT-OCSVM system categorizes alarms according to the level of severity they have, with most being classified as possible and those few originating from real attacks in the system are termed severe (see Fig. 14). Since the proposed mechanism is part of a distributed PIDS, the information sent by the IT-OCSVM can be combined with those sent by the other detection modules. For this reason this categorization of the alarms is important, if correct final decisions about the situation in the network are to be taken.

4.5. *Computational cost and time overhead*

The complexity of an IDS can be attributed to hardware, software and operational factors. For simplicity, it is usually estimated as the computing time required to perform classification of the dataset and output the final alarms. Increasing the number of classifiers usually increases the computational cost and decreases their comprehensibility. For this

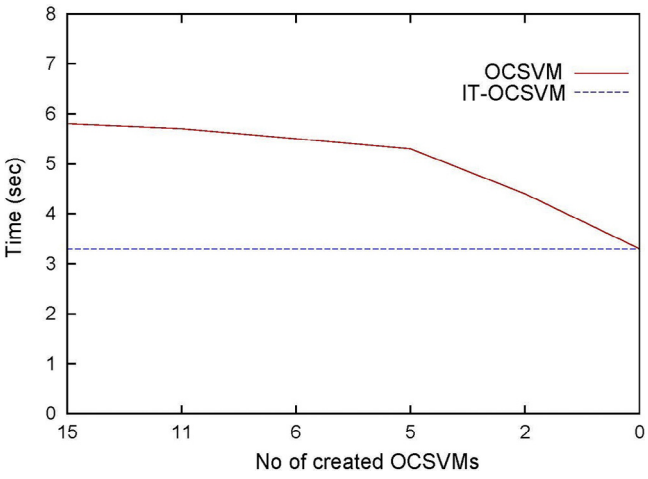

Fig. 15 – Approximate execution time for the entire testing dataset.

reason, special care must be taken when choosing parameter $P_{packets}$. As mentioned in Subsection 4.3, this parameter determines the number of created split datasets and thus split OCSVMs. While the increase in number of split OCSVMs does not impose any significant increase in the method's performance, this may slow down the detection mechanism.

In Fig. 15, we illustrate the time performance of the method compared to a simple OCSVM. The evaluation was conducted on a PC with Intel core 2 duo 1.7 MHz CPU, 2 GB main memory, 80 GB hard disk 7200 rpm hard disk and Microsoft windows 7 64 bit.

According to Fig. 15, the execution time of the proposed IT-OCSVM is bigger compared to a simple OCSVM method. However, or the extreme configuration where 15 OCSVMs are created, the performance gap increases and the proposed IT-OCSVM is 75% slower than the simple OCSVM, whereas when the IT-OCSVM operates under the default configuration (5–6 OCSVMs), the performance gap is 55%. Based on these observations we conclude that the system, under the default

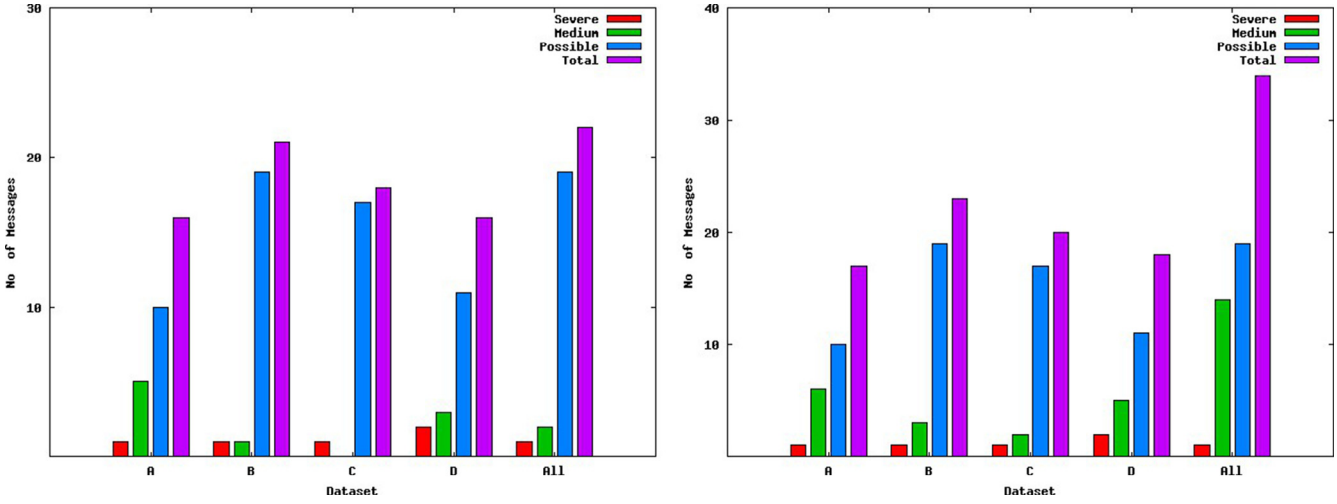

Fig. 14 – IT-OCSVM categorizes aggregated alarms. The left diagram shows aggregated alarms created by IT-OCSVM without the additional medium alarms and the right diagram illustrates all the final alarms created by the IT-OCSVM.

configuration, performs a classification in a comparable time to that of a simple OCSVM classifier, and thus, it can be adopted in soft real-time applications.

We have to mention that the performance evaluation conducted in this subsection does not include the time that each detection mechanism needs in order to create and disseminate IDMEF messages or the computational and time costs that the correlator needs in order to collect and analyze the alarms. It is evident that the OCSVM classifier, compared to the proposed IT-OCSVM, needs significant additional time in order to send all the detected alarms. Moreover, the categorization that is performed by the IT-OCSVM mechanism reduces the computational load of the correlator that collects the alarms from the distributed detection agents.

5. Conclusion

This article analyzes the IT-OCSVM mechanism and evaluates it against the baseline method for different attack scenarios. The detection mechanism, which runs in a distributed way, can be used in large SCADA networks with no additional modifications. The combination of social network analysis metrics with machine learning classification techniques enhances the performance of the detection mechanism and improves accuracy for all the simulation scenarios investigated. Moreover, the aggregation procedure embedded in the proposed mechanism decreases the overhead of the IT-OCSVM and makes it easily incorporable in a soft real time system. That is, the method produces a small amount of final alerts and manages to detect all the simulated attacks.

In future work the proposed mechanism will be further enhanced in order to decrease false alarms and increase detection accuracy. It will be tested in a bigger hybrid testbed under different attack scenarios and other behavior patterns will be also evaluated, e.g. the patterns of interaction among entities, thereby adding more sophistication to the detection mechanism.

REFERENCES

Balasubramaniyan JS, Garcia-Fernandez JO, Isacoff D, Spafford E, Zamboni D. An architecture for intrusion detection using autonomous agents. In: Proceedings of the 14th annual computer security applications conference. IEEE; 1998. pp. 13–24.

Bennett C, Highfill D. Networking AMI smart meters. In: IEEE energy 2030 conference, 2008 (ENERGY 2008); 2008. p. 1–8.

Chawla NV, Hall LO, Bowyer KW, Kegelmeyer WP. Learning ensembles from bites: a scalable and accurate approach. J Mach Learn Res 2004;5:421–51.

Crosbie M, Spafford EH. Active defense of a computer system using autonomous agents. 1995.

Curiac D-I, Volosencu C. Ensemble based sensing anomaly detection in wireless sensor networks. Expert Syst Appl 2012;39(10):9087–96.

Debar H, Curry DA, Feinstein BS. The intrusion detection message exchange format (IDMEF). 2007.

García-Teodoro P, Díaz-Verdejo J, Maciá-Fernández G, Vázquez E. Anomaly-based network intrusion detection: techniques, systems and challenges. Comput Secur 2009;28(1–2):18–28.

Gogoi P, Bhattacharyya D, Borah B, Kalita JK. A survey of outlier detection methods in network anomaly identification. Comput J 2011;54(4):570–88.

Hamid MA, Islam MS, Hong CS. Misbehavior detection in wireless mesh networks. In: 10th international conference on advanced communication technology, 2008 (ICACT 2008), vol. 2. IEEE; 2008. p. 1167–69.

Jiang J, Yasakethu L. Anomaly detection via one class SVM for protection of SCADA systems. In: International conference on cyber-enabled distributed computing and knowledge discovery (CyberC). IEEE; 2013. p. 82–88.

Kim M-J, Kang D-K. Ensemble with neural networks for bankruptcy prediction. Expert Syst Appl 2010;37(4): 3373–9.

Kołaczek G, Prusiewicz A. Social network approach to anomaly detection in network systems. Intrusion Detection Systems; 2011.

Krawczyk B, Woźniak M. Diversity measures for one-class classifier ensembles. Neurocomputing 2014;126:36–44.

Kumar S. Classification and detection of computer intrusions [Ph.D. thesis]. Purdue University; 1995.

Lazarevic A, Ertöz L, Kumar V, Ozgur A, Srivastava J. A comparative study of anomaly detection schemes in network intrusion detection. In: SDM. SIAM; 2003. p. 25–36.

Maglaras L, Jiang J. Intrusion detection in SCADA systems using machine learning techniques. In: Proceedings of the SAI conference; 2014. p. 626–31.

Maglaras LA, Jiang J, Cruz TJ. Integrated OCSVM mechanism for intrusion detection in SCADA systems. Electron Lett 2014;50:1935–6.

McDaniel P, McLaughlin S. Security and privacy challenges in the smart grid. IEEE Secur Priv 2009;7(3):75–7.

Menahem E, Rokach L, Elovici Y. Combining one-class classifiers via meta learning. In: Proceedings of the 22nd ACM international conference on conference on information & knowledge management. ACM; 2013. p. 2435–40.

Rokach L. Pattern classification using ensemble methods, vol. 75. World Scientific; 2010.

Shieh AD, Kamm DF. Ensembles of one class support vector machines. In: Multiple classifier systems. Springer; 2009. p. 181–90.

Spearman C. The proof and measurement of association between two things. Am J Psychol 1904;15(1):72–101.

Tax DM, Duin RP. Combining one-class classifiers. In: Multiple classifier systems. Springer; 2001. p. 299–308.

Tsoumakas G, Katakis I, Vlahavas I. Effective voting of heterogeneous classifiers. In: Machine learning: ECML 2004. Springer; 2004. p. 465–76.

Tumer K, Ghosh J. Error correlation and error reduction in ensemble classifiers. Connect Sci 1996;8(3–4):385–404.

Wang Y, Wong J, Miner A. Anomaly intrusion detection using one class SVM. In: Proceedings from the fifth annual IEEE SMC information assurance workshop, 2004. IEEE; 2004. p. 358–64.

Zhu B, Joseph A, Sastry S. A taxonomy of cyber attacks on SCADA systems. In: 2011 international conference on Internet of things (iThings/CPSCom) and 4th international conference on cyber, physical and social computing. IEEE; 2011. p. 380–88.

EAI Endorsed Transactions
on Security and Safety

Research Article ICST.ORG

A novel intrusion detection method based on OCSVM and K-means recursive clustering★

Leandros A. Maglaras[1,*], Jianmin Jiang[1]

[1]University of Surrey, Department of Computing, Guildford, UK

Abstract

In this paper we present an intrusion detection module capable of detecting malicious network traffic in a SCADA (Supervisory Control and Data Acquisition) system, based on the combination of One-Class Support Vector Machine (OCSVM) with RBF kernel and recursive k-means clustering. Important parameters of OCSVM, such as Gaussian width σ and parameter ν affect the performance of the classifier. Tuning of these parameters is of great importance in order to avoid false positives and over fitting. The combination of OCSVM with recursive k-means clustering leads the proposed intrusion detection module to distinguish real alarms from possible attacks regardless of the values of parameters σ and ν, making it ideal for real-time intrusion detection mechanisms for SCADA systems. Extensive simulations have been conducted with datasets extracted from small and medium sized HTB SCADA testbeds, in order to compare the accuracy, false alarm rate and execution time against the base line OCSVM method.

Keywords: Cyber security, SCADA systems, support vector machine, machine learning

Received on 27 October 2014; accepted on 30 October 2014, published on 30 January 2015

doi:10.4108/sesa.2.3.e5

1. Introduction

Several techniques and algorithms have been reported by researchers for intrusion detection [2, 3]. One big family of intrusion detection algorithms is rule based algorithms [4]. In real applications though, during abnormal situations, the behavior of the system cannot be predicted and does not follow any known pattern or rule. This characteristic makes rule based algorithms incapable of detecting the intrusion.

Generally, anomaly detection can be regarded as binary classification problem and thus many classification algorithms which are utilized for detecting anomalies, such as neural networks, support vector machines, K-nearest neighbor (KNN) and Hidden Markov model can be used. However, strictly speaking, they are not intrusion detection algorithms, as they require knowing what kind of anomaly is expecting, which deviates the fundamental object of intrusion detection. In addition these algorithms may be sensitive to noise in the training samples.

Segmentation and clustering algorithms [5] seem to be better choices because they do not need to know the signatures of the series. The shortages of such algorithms are that they always need parameters to specify a proper number of segmentation or clusters and the detection procedure has to shift from one state to another state. Negative selection algorithms [6] on the other hand,are designed for one-class classification; however, these algorithms can potentially fail with the increasing diversity of normal set and they are not meant to the problem with a small number of self-samples, or general classification problem where probability distribution plays a crucial role. Furthermore, negative selection only works for a standard sequence, which is not suitable for on line detection. Other algorithms, such as time series analysis are also introduced to anomaly detections, and again, they may not be suitable for most of the real application cases.

To minimize the above mention drawbacks an intelligent approach based on OCSVM [One-Class Support Vector Machine] principles are proposed for intrusion detection. OCSVM is a natural extension of the support vector algorithm to the case of unlabeled data, especially for detection of outliers. The OCSVM algorithm maps input data into a high dimensional feature space (via a kernel) and iteratively finds the maximal margin hyperplane which best separates the training data from the origin (Figure 1).

OCSVM principles have shown great potential in the area of anomaly detection [7–9]. IDS can provide active detection and automated responses during intrusions [10]. Commercial IDS products such as NetRanger, RealSecure, and Omniguard Intruder alert work on attack signatures. These signatures needed to be updated by the vendors on a regular basis in order to protect from new types of attacks. Most of the current intrusion

★This is an extended version of the paper [1]
*Corresponding author. Email: l.maglaras@surrey.ac.uk

detection commercial softwares are based on approaches with statistics embedded feature processing, time series analysis and pattern recognition techniques. Several extensions of OCSVM method have been introduced lately [11, 12].

OCSVM similar to other one-class classifiers e.g. GDE[13], PGA [14], suffer from false positive and over fitting situations. Intrusion detection systems (IDS) fail to deal with all kinds of attacks, while on the other hand, false alarms that are arisen from high sensitive IDS arise high economic risks. These situations are described in subsection 1.1

1.1. motivation

For the OCSVM with an RBF kernel, two parameters σ and ν need to be carefully selected in order to obtain the optimal classification result. A common strategy is to separate the data set into two parts, of which one is considered unknown. The prediction accuracy obtained from the unknown set more precisely reflects the performance on classifying an independent data set. An improved version of this procedure is known as cross-validation. Cross-validation is a model validation technique for assessing how the results of a statistical analysis will generalize to an independent data set. It is mainly used in settings where the goal is prediction, and one wants to estimate how accurately a predictive model will perform in practice.

In ν-fold cross-validation [15, 16], the training set is divided into ν subsets of equal size. Sequentially one subset is tested using the classifier trained on the remaining ν .. 1 subsets . Thus, each instance of the whole training set is predicted once so the cross-validation accuracy is the percentage of data which are correctly classified. The cross-validation procedure can prevent the over fitting problem.

Using an ensemble of decision mechanisms with different parameters, is another method to have an optimal result. An ensemble of classifiers [17] is a set of classifiers whose individual decisions are combined in some way. more trusted final decision. Ensemble systems of classifiers are widely used for intrusion detection in networks. Classifier ensemble design aims to include mutually complementary individual classifiers which are characterized by high diversity either in terms of classifier structure [18], internal parameters [19] or classifier inputs [20].

Unnthorsson et al. [21] proposed another method to select parameters for the OCSVM. In their method, ν was first set to a user-specified allowable fraction of misclassification of the target class (e.g. 1% or 5%), then the appropriate σ value was selected as the value for the classification accuracy curve of training samples first reaches $1 - \nu$. The obtained ν and σ combination can then be used in the OCSVM classification.

OCSVM similar to other one-class classifiers suffer from false positive and over fitting. The former is a situation that occurs when the classifier fires an alarm in the absence of real anomaly in the system and happens when parameter σ has too large vale. The latter is the situation when a model begins to memorize training data rather than learning to generalize from trend and it shows up when parameter σ is given relatively small value [22].

In this article we propose the combination of OCSVM method with a recursive k-means clustering, separating the real from false alarms in real time and with no pre-selection of parameters σ and ν.

1.2. Contributions

The present article develops a intrusion detection method, namely the *K-means OCSVM* ($\mathcal{K}-\mathcal{OCSVM}$). Using the well known OCSVM method with default values for parameters σ and ν we distinguish real from false alarms with the use of a recursive k-means clustering method. This is very different from all previous methods that required pre-selection of parameters with the use of cross-validation or other methods that ensemble outcomes of One class classifiers.

The article makes the following contributions:

- A new one class classifier $\mathcal{K}-\mathcal{OCSVM}$ is proposed.
- The proposed classifier combines a OCSVM with RBF kernel with a recursive K-means clustering method.
- $\mathcal{K}-\mathcal{OCSVM}$ separates in real time false from real alarms.
- A performance evaluation of the proposed method against a OCSVM classifier with different parameters is conducted, which attest the stability of the new structure.

The rest of this article is organized as follows: Section 2 describes the OCSVM method. In Section 3 the use of OCSVM in SCADA systems is presented; Section 4 presents the features extracted from the network traces for the testing and training of the model Section 5 describes the $\mathcal{K}-\mathcal{OCSVM}$ method; Section 6 presents the simulation environment and results. In Section 7 discusses possible enhancements of the method. Section 8 concludes the article.

2. OCSVM method

The one-class classification problem is a special case of the conventional two-class classification problem, where only data from one specific class are available and well represented. This class is called the target class. Another class, which is called the outlier class, can be sampled very sparsely, or even not at all. This smaller

class contains data that appear when the operation of the system varies from the normal, due to a possible attack. In general cases, the outlier class might be very difficult or expensive to measure. Therefore, in the one class classifier training process, mainly samples from the target class are used and there is no information about its counterpart. The boundary between the two classes has to be estimated from data in the only available target class. Thus, the task is to define a boundary around the target class, such that it encircles as many target examples as possible and minimizes the chance of accepting outliers.

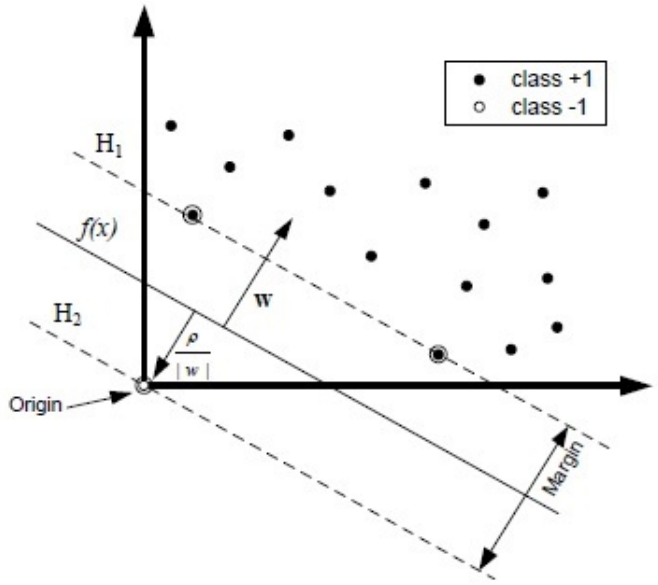

Figure 1. OCSVM maps input data into a high dimensional feature space

Scholkopf et al. [23] developed an OCSVM algorithm to deal with the one-class classification problem. The OCSVM may be viewed as a regular two-class SVM, where all the training data lie in the first class, and the origin is taken as the only member of the second class. The OCSVM algorithm first maps input data into a high dimensional feature space via a kernel function and then iteratively finds the maximal margin hyperplane, which best separates the training data from the origin. Thus, the hyperplane (or linear decision boundary) corresponds to the classification function

$$f(x) =< x, w > +b \tag{1}$$

where w is the normal vector and b is a bias term. The OCSVM solves an optimization problem to find the function f with maximal geometric margin. This classification function can be used to assign a label to a test example x. If $f(x) < 0$, then x is labeled as an anomaly (outlier class), otherwise it is labeled normal (target class).

Using kernel functions, solving the OCSVM optimization problem is equivalent to solving the following dual quadratic programming problem.

$$min\frac{1}{2}\sum_{i,j} a_i a_j K(x_i, x_j) \tag{2}$$

subject to $0 \leq a_i \leq 1/\nu l$, and $\sum_i a_i \leq 1$.

Parameter a_i is a Lagrange multiplier, which can be thought of as a weight for example x, such that vectors associated with non-zero weights are called support vectors and solely determine the optimal hyperplane, ν is parameter that controls the trade-off between maximizing the number of data points contained by the hyperplane and the distance of the hyperplane from the origin, l is the number of points in the training dataset, and $K(x_i, x_j)$ is the kernel function.

Using the kernel function to project input vectors into a feature space, nonlinear decision boundaries are allowed. Generally, four types of kernel are often used: linear, polynomial, sigmoid and Gaussian radial basis function (RBF) kernels. In this paper, we use the RBF kernel, which has been commonly used for the OCSVM.

$$K(x_i, x_j) = exp(-\sigma||x_i - x_j||^2), \quad \sigma > 0 \tag{3}$$

Although the OCSVM requires samples of the target class only as training samples, some studies showed that when negative examples (i.e. samples of outlier classes) are available, they can be used during the training to improve the performance of the OCSVM. In this paper only normal data were used for the training of the method, though a similar to the one proposed by Tax [24], which includes a small amount of samples of the outlier class, will be also applied and evaluated in the near future.

3. OCSVM for SCADA system

Cyber-attacks against SCADA systems [25] are considered extremely dangerous for Critical Infrastructure (CI) operation and must be addressed in a specific way [26]. Presently one of the most adopted attacks to a SCADA system is based on fake commands sent from the SCADA to the RTUs. OCSVM [27, 28] possesses several advantages for processing SCADA environment data and automate SCADA performance monitoring, which can be highlighted as:

- In the case of SCADA performance monitoring, which patterns in data are normal or abnormal may not be obvious to operators. Since OCSVM does not require any signatures of data to build the detection model it is well suited for intrusion detection in SCADA environment.

- Since the detection mechanism does not require any prior information of the expected attack types, OCSVM is capable of detection both known and unknown (novel) attacks.
- In practice training data, taken from SCADA environment, could include noise samples. Most of the classification based intrusion detection methods are very sensitive to noise. However, OCSVM detection approach is robust to noise samples in the training process.
- Algorithm configuration can be controlled by the user to regulate the percentage of anomalies expected.
- Due to the low computation time, OCSVM detectors can operate fast enough for online SCADA performance monitoring.
- Typically monitoring data of SCADA systems consists of several attributes and OCSVM is capable of handling multiple attributed data.

4. Attribute extraction

Feature extraction is essential in a classification problem. In order to train the OCSVM module properly we used network trace files from a secure wireless network in University of Surrey. Based on previous analysis of data [29] we selected some initial features that are used as attributes for our OCSVM model .

Attributes in the network traces datasets have all forms: continuous, discrete, and symbolic, with significantly varying resolution and ranges. Most pattern classification methods are not able to process data in such a format. Hence pre-processing was required before pattern classification models could be built. Pre-processing consists of two steps: first step involved mapping symbolic-valued attributes to numeric-valued attributes and second step implemented scaling The main advantage of scaling is to avoid attributes in greater numeric ranges dominating those in smaller numeric ranges. Another advantage is to avoid numerical difficulties during the calculation.

Based on the above observations we used the network trace dataset in order to test our OCSVM module. The attributes that we used for this initial training / testing phase, were rate & packet size. The values were scaled to the range [0,1].

The rate (1^{st} attribute) was calculated using the equation 4:

$$Rate_{scaled} = \frac{Time\ difference}{Max\ time\ difference} \tag{4}$$

Time difference is calculated by the difference of time of current packet and the time of previous packet injected in the system.

The packet size (2^{nd} attribute) was scaled using the equation 5:

$$Packet_{scaled} = \frac{packet\ size}{Max\ packet\ size} \tag{5}$$

5. $\mathcal{K}-\mathcal{OCSVM}$

The proposed $\mathcal{K}-\mathcal{OCSVM}$ combines the well known OCSVM classifier with the RBF kernel with a recursive K-means clustering module. Figure 2 illustrates the procedure of intrusion detection of our proposed $\mathcal{K}-\mathcal{OCSVM}$ model.

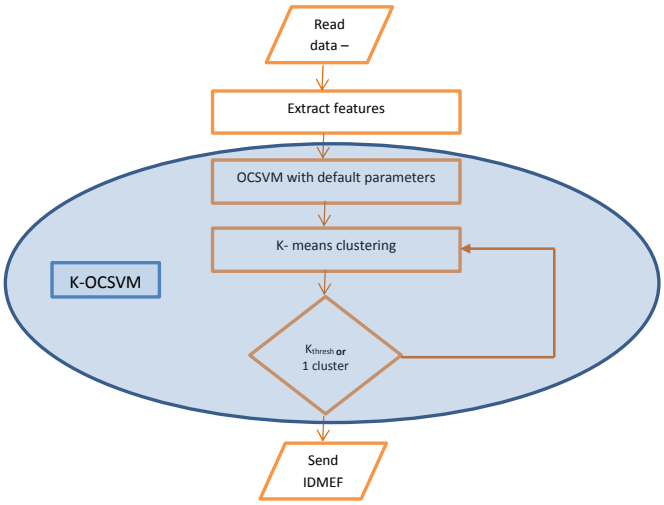

Figure 2. $\mathcal{K}-\mathcal{OCSVM}$ module

The OCSVM classifier runs with default parameters and the outcome consists of all possible outliers. These outliers are clustered using the k-means clustering method with *2* clusters, where the initial means of the clusters are the maximum and the minimum negative values returned by the OCSVM module. From the two clusters that are created from the K-means clustering, the one that is closer to the maximum negative value (severe alerts) is used as input in the next call of the K-means clustering. This procedure is repeated until all outcomes are put in the same cluster or the divided set is big enough compared to the initial one, according to the threshold parameter k_{thres}.

K-means clustering method divides the outcomes according to their values and those outcomes with most negative values are kept. That way, after the completion of this recursive procedure only the most severe alerts are communicated from the $\mathcal{K}-\mathcal{OCSVM}$. The division of the data need no previous knowledge about the values of the outcomes which may vary from *-0.1* to *-160* depending of the assigned values to parameters σ and ν. The method can find the most important/possible outliers for any given values to parameters σ and ν.

One important parameter that affects the performance of $\mathcal{K}-\mathcal{OCSVM}$ is the value of threshold k_{thres}. For given value *2*, the final cluster of severe alerts that the method communicates to other parts of the IDS system is limited to *2* to *4* alarms. For bigger value (*3* or more) the number of alerts also rises till the method degrades to the initial OCSVM. The optimal value for the given parameter k_{thres} is a matter for future investigation.

In order to cooperate with the other modules the OCSVM module needed to be integrated in the *PID* system and communicate with the other modules. Once an intrusion is detected several actions can be taken by the *IDS* (intrusion detection system. These actions include recording of intrusions in log files, sending of alert messages, limit the bandwidth of the intruder or even block all connections from the intruder. In order to better cooperate with the other components/modules that are being produced in the *CockpitCI* project the OCSVM model sends IDMEF [30] files.

6. Performance evaluation

6.1. Training of OCSVM model

The initial training of both the OCSVM and the $\mathcal{K}-\mathcal{OCSVM}$ modules is conducted using several trace files

- a trace file that is sniffed out of a typical wireless network (Figure 3) that consists of *10.000* lines each representing a packet send in the network.
- datasets of a testbed under normal operation
- datasets of a medium sized SCADA system under normal operation

To train the OCSVM, we adopt the *RBF* for the kernel equation. This kernel nonlinearly maps samples into a higher dimensional space so it can handle the case when the relation between class labels and attributes is nonlinear.

```
1 1: 0 2: 0.101694915254237
1 1: 1.08894782018269E-05 2: 0.101694915254237
1 1: 0.318975236045454 2: 0.108474576271186
1 1: 0.000146687676954043 2: 0.108474576271186
1 1: 1 2: 0.101694915254237
```

Figure 3. Format of the transformed Network trace file

The training model that is extracted after the training of the OCSVM is used for on line detection of malicious data. Since the model is based on features that are related to network traffic, and since the traffic of the system varies from area to area and from time period to time period, possible generation of multiple models could improve the performance of the module.

The network traffic in electric grids varies according to the activity which is not constant during the day. Also in some areas the activity follows different patterns according to the local demand. These characteristics maybe be critical for the proper training of the module and the accurate detection of intruders.

6.2. Testing of OCSVM model

We evaluate the performance of the method using data from the wireless network of the University campus, from a testbed that mimics a small-scale SCADA system and from a Hybrid testbed of a medium sized SCADA system. The parameters used for the evaluation of the performance of $\mathcal{K}-\mathcal{OCSVM}$ are listed in Table 1.

Parameter	*Range of Values*	*Default value*
σ	0.1 - 0.0001	0.007
ν	0.002 - 0.05	0.01
k_{thresh}	2 - 3	2

Table 1. Evaluation Parameters

Wireless network. In order to test our model we use another network trace file sniffed from the wireless network. The testing trace file consists of it 30.000 lines. We compare the performance of our proposed model against OCSVM classifiers having the same values for parameters σ and ν. We name each OCSVM classifier according to the parameters σ and ν : $OCSVM_{0.07,0.01}$ stands for OCSVM classifier with parameters $\sigma = 0.07$ and $\nu = 0.01$.

In Table 2 we show the number of observed anomalies detected from OCSVM and $\mathcal{K}-\mathcal{OCSVM}$ respectively. From this table it is shown how parameters σ, ν affect the performance of OCSVM. Even for a value of ν equal to *0.005*, OCSVM produces over *400* possible attacks, making the method inappropriate for a SCADA system where each false alarm is costly.

Parameter σ	*Parameter* ν	$\mathcal{K}-\mathcal{OCSVM}$	*ocsvm*
0.007	0.002	3	408
0.007	0.01	3	299
0.007	0.005	2	408
0.0001	0.01	3	274
0.1	0.01	2	295

Table 2. Performance evaluation of $\mathcal{K}-\mathcal{OCSVM}$ and OCSVM for $K_{thers} = 2$.

In figure 4 we present the outcome that OCSVM produces for the training network trace under different values of parameters σ and ν. From this figure it is obvious that the outcome is strongly affected by

the values of these parameters, making $\mathcal{K}-\mathcal{OCSVM}$ necessary tool for proper intrusion detection.

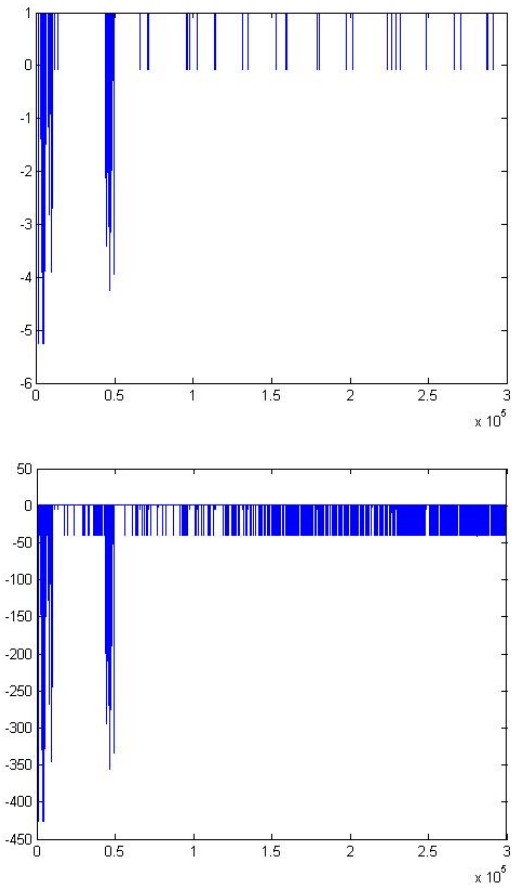

Figure 4. OCSVM classification outcome for different values of parameters σ, ν Upper diagram : $OCSVM_{0.007,0.001}$, Lower diagram : $OCSVM_{0.01,0.05}$

Testbed scenario. The second trial is conducted off line with the use of two datasets extracted from the testbed (Figure 5). The testbed architecture mimics a small-scale SCADA system, comprising the operations and field networks and including a Human-Machine Interface Station (for process monitoring), a managed switch (with port monitoring capabilities, for network traffic capture), and two Programmable Logic Controller Units, for process control. The NIDS and OCSVM modules are co-located on the same host, being able to intercept all the traffic flowing on the network scopes.

During the testing period several attack scenarios are simulated in the testbed. These scenarios include network scan, network flood and MITM attack.

Three kinds of attacks are being evaluated:

- **Network scan attack** In typical network scan attack, the attacker uses TCP/FIN scan to determine if ports are closed to the target machine. Closed ports answer with RST packets while open ports discard the FIN message. FIN packets blend with background noise on a link and are hard to be detected.

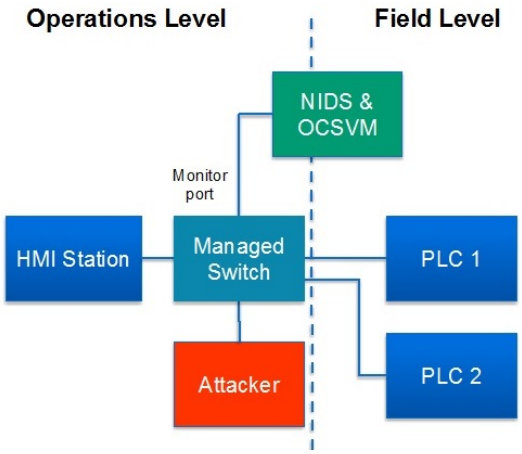

Figure 5. Architecture of the testbed

- **ARP cache spoofing - MITM attack ARP cache spoofing** is a technique where an attacker sends fake ARP messages. The aim is to associate the attacker's MAC address with the IP address of another host, causing any traffic meant for that IP to be sent to attacker instead. The attacker could choose to inspect the packets, modify data before forwarding (**man-in-the-middle attack**) or launch a denial of service attack by causing some of the packets to be dropped.

- **DoS attack** Network flood is the instance where the attacker floods the connection with the PLC by sending SYN packets. In a TCP SYN flooding attack, an attacker sends many SYN messages, with fictitious (spoofed) IP addresses, to a single node (victim). Although the node replies with SYN/ACK messages, these messages are never acknowledged by the client. As a result, many half open connections exist on the victim, consuming its resources. This continues until the victim has consumed all its resources, hence can no longer accept new TCP connection requests.

In Table 3 we show the number of alert messages (IDMEF) sent from OCSVM and $\mathcal{K}-\mathcal{OCSVM}$ respectively. From this table it is shown how parameters σ, ν affect the performance of OCSVM for the testbed scenario. While for the same network trace file OCSVM produces from *10529* to *10704* alert messages according to the values of the parameters, $\mathcal{K}-\mathcal{OCSVM}$ produces the same *120* alert messages. All the reported attacks are concerning the *DoS* attack that creates the biggest fluctuation in the network traffic.

Parameter σ	*Parameter* ν	$\mathcal{K}-\mathcal{OCSVM}$	*ocsvm*
0.007	0.002	120	10529
0.007	0.01	120	10703
0.007	0.005	120	10584
0.0001	0.01	120	10602
0.1	0.01	120	10704

Table 3. Performance evaluation of $\mathcal{K}-\mathcal{OCSVM}$ and OCSVM for $K_{thers} = 2$.

Testbed scenario with split testing periods. Since the attacks are performed during different time periods we divide the testing dataset in several smaller ones, each containing a different attack. Testing data consists of normal data and attack data and the composition of the data sets are as follows:

- Testing set-A' : *1 - 5000*: Normal data
- Testing set-B' : *5000 - 10000*: Normal data + **Arp spoofing** attack + **Network scan**
- Testing set-C' : *10000 - 25000*: Normal data + **Flooding Dos attack** + **Network scan**
- Testing set-D' : *25000 - 41000*: Normal data + **MITM attack**

Dataset	*Initial alarms*	*Aggregated alarms*
A	129	2
B	658	3
C	9273	120
D	203	3
All	10507	120

Table 4. Aggregated alarms produced by $\mathcal{K}-\mathcal{OCSVM}$ are significantly decreased compared to the initial alarms

From table 4 we observe that not only the most important intrusions are detected and reported but also the total overhead on the system is limited. For all time periods the messages communicated reflect actual attacks in the network, except from the testing set-A'. In this time period *HMI* station demonstrated a significant variation in the rate that it injected packets in the system between testing and training of the module. This is due to the limited training of the OCSVM and can be avoided if training dataset consists of data that represent the traffic in the network during under work loads. The increased number of alarms created from $\mathcal{K}-\mathcal{OCSVM}$ for the dataset B' is due to the fact in this time period the attacker uses an excessive number of SYN packets in order to flood the communication channel.

Hybrid Testbed scenario. The third trial is conducted off line with the use of large datasets extracted from a Hybrid Testbed (HTB) scenario. The Hybrid testbed architecture mimics a medium-scale SCADA system, comprising the operations and field networks and including Human-Machine Interface Stations (for process monitoring), six managed switches (with port monitoring capabilities, for network traffic capture), and several Programmable Logic Controller Units, for process control. The initial dataset consists of over *3* million rows, each representing a packet sent in the system, capturing network of several days. The dataset is split in *65* smaller ones of *50.000* rows. The datasets contain only data from a normal operation of the HTB.

Both OCSVM and $\mathcal{K}-\mathcal{OCSVM}$ are trained and tested with these datasets, using cross validation. The mean number of alert messages sent by the two modules is shown in Table 5.

Parameter σ	*Parameter* ν	$\mathcal{K}-\mathcal{OCSVM}$	*ocsvm*
0.007	0.002	1 - 2	40
0.007	0.01	1 - 2	207
0.007	0.005	1 - 2	105
0.0001	0.01	1 - 2	85
0.1	0.01	1 - 2	271

Table 5. Performance evaluation of $\mathcal{K}-\mathcal{OCSVM}$ and OCSVM for $K_{thers} = 2$

Using real datasets of a medium sized HTB SCADA system the performance of the proposed $\mathcal{K}-\mathcal{OCSVM}$ method is very stable compared to a simple OCSVM under the same configuration. This behavior is very promising since $\mathcal{K}-\mathcal{OCSVM}$ method has a very low false alarm rate (*lower than 0.02 %*) while on the same time the overhead induced by the method is negligible (C.F. Subsection 6.3. We must state here that for the HTB we had available only non malicious datasets for the evaluatioin of the proposed method. In the future when datasets containing malicious attacks are available an extensive evaluation of the $\mathcal{K}-\mathcal{OCSVM}$ method is going to be conducted in terms of accuracy and false alarm rate.

6.3. Computational cost and time overhead

Complexity of an intrusion detection system can be attributed to hardware, software and operation factors. For simplicity, it is usually estimated as the computing time required to perform classification of the dataset and output the final alarms. In order to evaluate the complexity of the proposed method we calculate the execution time and compare it to a simple OCSVM module. The evaluation is conducted on a PC with Intel

core *2* duo *1.7* Mhz CPU, *2*GB main memory, *80*GB hard disk *7200* rpm hard disk and Microsoft windows *7 64*bit. In Figure 6 we represent the time performance of the method compared to a simple OCSVM module for the testbed scenario.

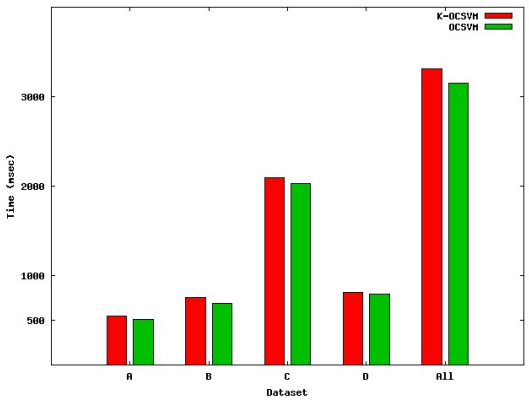

Figure 6. Computational cost for the testbed scenario

According to Figure 6 execution time of the proposed $\mathcal{K}-\mathcal{OCSVM}$ is slightly bigger compared to a simple OCSVM method. The performance gap is around *5%* to *10%* for all the datasets used in the simulation. Based on these observations we conclude that the system, performs a classiffication in a comparable time to that of a simple OCSVM classifier, and it thus can be adopted in soft real-time applications. We have to mention that the performance evaluation which is conducted in this subsection, does not include the time that each detection mechanism needs in order to create and disseminate IDMEF messages. It is evident that the OCSVM classifer, compared to the proposed $\mathcal{K}-\mathcal{OCSVM}$, needs significant additional time in order to send all the detected alarms.

7. Discussion

The proposed $\mathcal{K}-\mathcal{OCSVM}$ can significantly reduce the produced alarms from the OCSVM module that is the heart of the detection mechanism. The profound advantages of low overhead and low false alarm rate come with the cost of lower accuracy and higher computational overhead. In this section we discuss some enhancements of the proposed method that can improve its performance.

7.1. Parameter k_{thres}

As stated is Section 5 $\mathcal{K}-\mathcal{OCSVM}$ method is a recursive clustering of the alarms produced by the OCSVM module. This recursive procedure is used in order to distinguish severe from possible alarms and finally disseminate only those that represent an actual misbehavior of the system. This way the OCSVM module is enhanced in both the decreased overhead that induces to the system from the disseminated alarm files and in the decreased false alarm rate that it has. The recursive method stops when either all initial alarms are put in the same cluster or when the divided set is big enough compared to the initial one, according to the threshold parameter k_{thres}.

In the simulations presented in Subsection 6 the parameter is set to *2*. When raising the value of this parameter the produced final alarms of the proposed $\mathcal{K}-\mathcal{OCSVM}$ method raises. This raise also leads to a raise in the false alarm rate. On the other hand the accuracy of the method raises since less profound attacks are detected. By raising the value of the parameter above one limit the method degrades to the initial OCSVM. The optimum value for parameter $\mathcal{K}-\mathcal{OCSVM}$ varies according to the architecture of the network. For big disperse networks large value of the parameter would lead to the creation of too many alarms from the module, while on the same time in a medium sized network very small value of the parameter would lead to a dangerous decrease of the detection capabilities of the module.

Except from the static configuration of the network, traffic conditions can also affect the performance of the method. In real SCADA systems the network traffic varies between daytime and night, weekdays and weekends. In order to cope with both static and dynamic features of the network an enhanced $\mathcal{K}-\mathcal{OCSVM}$ method that dynamically adapts the parameter k_{thres}, similar to [31, 32] could be effective.

7.2. Multi stage $\mathcal{K}-\mathcal{OCSVM}$

The proposed method uses only the values of the initial alarms in order to filter out those that don't represent an actual attack. That way attacks that cause significant variation in certain features of the OCSVM module are detected, while on the same time other more insidious attacks are passing undetected. A multi stage $\mathcal{K}-\mathcal{OCSVM}$ where both the number of attacks that share common characteristics, like origin, destination, port number e.t.c., and the actual values of the attacks can be developed in order to better detect different kinds of attacks.

Figure 7 represents a possible architecture of a two stage $\mathcal{K}-\mathcal{OCSVM}$ module. The number of the stages can be increased and the alarms produced by each stage can be further aggregated. The fusion of the outputs of the different stages can be done using any of the existing ensemble methods .i.e. majority voting, performance weighting, distribution summation, order statistics e.t.c.

8. Conclusions

We have presents an intrusion detection module for SCADA systems that is based in OCSVM classifier and

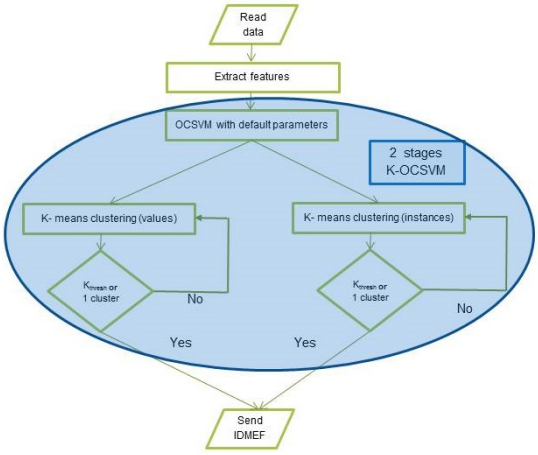

Figure 7. Architecture of a Multi stage $\mathcal{K}-\mathcal{OCSVM}$

a recursive k-means clustering method. The module is trained off-line by network traces, after the attributes are extracted from the network dataset. The intrusion detection module is part of an distributed IDS system.

The method is tested on three different datasets. For the first testing scenario, traces of a wireless network are used. This test shows that the method is stable and its performance is not influenced by the selection of parameters ν and σ. For the second scenario, testing of the proposed module is conducted with datasets that are sniffed of a testbed that mimics a small-scale SCADA system under different attack scenarios. After the completion of the test, not only the most important intrusions are detected and reported by $\mathcal{K}-\mathcal{OCSVM}$ but also the total overhead on the system is limited. Finally extensive testing of the $\mathcal{K}-\mathcal{OCSVM}$ module with real datasets extracted from a medium sized HTB SCADA system shows that the performance of the proposed $\mathcal{K}-\mathcal{OCSVM}$ method remains very stable under different configurations. After the execution of the $\mathcal{K}-\mathcal{OCSVM}$ method, for all the simulated scenarios, only severe alerts are communicated to the system by IDMEF files that contain information about the source, destination, protocol and time of the intrusion.

The main feature of $\mathcal{K}-\mathcal{OCSVM}$ module is that it can perform anomaly detection in a time-efficient way, with good accuracy and low overhead. Low overhead is an important evaluation metric of a distributed detection module that is scattered in a real-time system, since frequent communication of IDMEF files from detection agents degrade the performance of the SCADA network. Recursive k-means clustering, reassures that small fluctuations on network traffic, which most of the times cause OCSVM to trigger false alarms, are ignored by the proposed detection module. The added computational time of the method compared to a simple OCSVM varies between 5% and 10% which results in a neglective time overhead. This overhead does not include the time that each detection mechanism needs in order to create and disseminate IDMEF messages. By adding the time needed in order to create and send each IDMEF file to the IDS management system the proposed $\mathcal{K}-\mathcal{OCSVM}$ method prevails on the overall performance in terms of time efficiency.

Finally we investigate how parameter k_{thres} affects the performance of the method and proposed a multi-stage $\mathcal{K}-\mathcal{OCSVM}$ method for better accuracy. As future work we will conduct an in depth performance evaluation of the proposed mechanism. Using malicious and attack-free datasets of the HTB SCADA testbed, we are going to evaluate $\mathcal{K}-\mathcal{OCSVM}$'s performance in terms of false positive rate and accuracy. Using the evaluation outcomes we are planning to modify the proposed $\mathcal{K}-\mathcal{OCSVM}$ in order to further decrease false alarms and improve overall performance.

Acknowledgement. Research supported by the project "CockpitCI: Cybersecurity on SCADA: risk prediction, analysis and reaction tools for Critical Infrastructures", funded under European Framework-7 Programme (contract No. 285647)

References

[1] Maglaras, L.A. and Jiang, J. (2014) Ocsvm model combined with k-means recursive clustering for intrusion detection in scada systems. In *Proceedings of the 10th Qshine conference* (EAI).

[2] Khan, L., Awad, M. and Thuraisingham, B. (2007) A new intrusion detection system using support vector machines and hierarchical clustering. *The VLDB JournalThe International Journal on Very Large Data Bases* **16**(4): 507–521.

[3] Mukkamala, S., Janoski, G. and Sung, A. (2002) Intrusion detection using neural networks and support vector machines. In *Neural Networks, 2002. IJCNN'02. Proceedings of the 2002 International Joint Conference on* (IEEE), **2**: 1702–1707.

[4] Roesch, M. *et al.* (1999) Snort: Lightweight intrusion detection for networks. In *LISA*, **99**: 229–238.

[5] Portnoy, L. (2000) Intrusion detection with unlabeled data using clustering .

[6] Kim, J. and Bentley, P.J. (2001) An evaluation of negative selection in an artificial immune system for network intrusion detection. In *Proceedings of GECCO*: 1330–1337.

[7] Wang, Y., Wong, J. and Miner, A. (2004) Anomaly intrusion detection using one class svm. In *Information Assurance Workshop, 2004. Proceedings from the Fifth Annual IEEE SMC* (IEEE): 358–364.

[8] Ma, J. and Perkins, S. (2003) Time-series novelty detection using one-class support vector machines. In *Neural Networks, 2003. Proceedings of the International Joint Conference on*, **3**: 1741–1745 vol.3.

[9] Li, K.L., Huang, H.K., Tian, S.F. and Xu, W. (2003) Improving one-class svm for anomaly detection. In

Machine Learning and Cybernetics, 2003 International Conference on (IEEE), **5**: 3077–3081.

[10] Dasgupta, D. and Gonzalez, F.A. (2001) An intelligent decision support system for intrusion detection and response. In *Information Assurance in Computer Networks* (Springer), 1–14.

[11] Glazer, A., Michael, L. and Markovitch, S. (2013) q-ocsvm: A q-quantile estimator for high-dimensional distributions. In *In Proceedings of The 27th Conference on Neural Information Processing Systems (NIPS-2013),Lake Tahoe, Nevada*.

[12] Song, X., Fan, G. and Rao, M. (2008) Svm-based data editing for enhanced one-class classification of remotely sensed imagery. *Geoscience and Remote Sensing Letters, IEEE* **5**(2): 189–193.

[13] Eskin, E., Arnold, A., Prerau, M., Portnoy, L. and Stolfo, S. (2002) A geometric framework for unsupervised anomaly detection. In *Applications of data mining in computer security* (Springer), 77–101.

[14] Knorr, E.M. and Ng, R.T. (1997) A unified notion of outliers: Properties and computation. In *KDD*: 219–222.

[15] Burman, P. (1989) A comparative study of ordinary cross-validation, v-fold cross-validation and the repeated learning-testing methods. *Biometrika* **76**(3): 503–514.

[16] Hastie, T., Tibshirani, R., Friedman, J., Hastie, T., Friedman, J. and Tibshirani, R. (2009) *The elements of statistical learning*, **2** (Springer).

[17] Menahem, E., Rokach, L. and Elovici, Y. (2013) Combining one-class classifiers via meta learning. In *Proceedings of the 22nd ACM international conference on Conference on information & knowledge management* (ACM): 2435–2440.

[18] Tsoumakas, G., Katakis, I. and Vlahavas, I. (2004) Effective voting of heterogeneous classifiers. In *Machine Learning: ECML 2004* (Springer), 465–476.

[19] Kim, M.J. and Kang, D.K. (2010) Ensemble with neural networks for bankruptcy prediction. *Expert Systems with Applications* **37**(4): 3373–3379.

[20] Krawczyk, B. and Woźniak, M. (2014) Diversity measures for one-class classifier ensembles. *Neurocomputing* **126**: 36–44.

[21] Runarsson, T.P. and Jonsson, M.T. (2003) Model selection in one-class ν-svms using rbf kernels. In *Proceedings of 16th International Congress and Exhibition on Condition Monitoring and Diagnostic Engineering Management*.

[22] Li, X., Wang, L. and Sung, E. (2008) Adaboost with svm-based component classifiers. *Engineering Applications of Artificial Intelligence* **21**(5): 785–795.

[23] Schölkopf, B., Platt, J.C., Shawe-Taylor, J., Smola, A.J. and Williamson, R.C. (2001) Estimating the support of a high-dimensional distribution. *Neural computation* **13**(7): 1443–1471.

[24] TAX, D. (2001) One-class classification. *PhD thesis,Delft University of Technology, The Netherlands* .

[25] Barbosa, R.R.R. and Pras, A. (2010) Intrusion detection in scada networks. In *Mechanisms for Autonomous Management of Networks and Services* (Springer), 163–166.

[26] Zhu, B., Joseph, A. and Sastry, S. (2011) A taxonomy of cyber attacks on scada systems. In *Internet of Things (iThings/CPSCom), 2011 International Conference on and 4th International Conference on Cyber, Physical and Social Computing* (IEEE): 380–388.

[27] Jiang, J. and Yasakethu, L. (2013) Anomaly detection via one class svm for protection of scada systems. In *Cyber-Enabled Distributed Computing and Knowledge Discovery (CyberC), 2013 International Conference on* (IEEE): 82–88.

[28] Zhang, R., Zhang, S., Lan, Y. and Jiang, J. (2008) Network anomaly detection using one class support vector machine. In *Proceedings of the International MultiConference of Engineers and Computer Scientists*, **1**.

[29] Maglaras, L.A. and Jiang, J. (2014) Intrusion detection in scada systems using machine learning techniques. In *Proceedings of the 2nd SAI conference* (SAI).

[30] Debar, H., Curry, D.A. and Feinstein, B.S. (2007) The intrusion detection message exchange format (idmef) .

[31] Campbell, C., Cristianini, N. and Shawe-Taylor, J. (1999) Dynamically adapting kernels in support vector machines. *Advances in neural information processing systems* **11**: 204–210.

[32] Cao, L. and Gu, Q. (2002) Dynamic support vector machines for non-stationary time series forecasting. *Intelligent Data Analysis* **6**(1): 67–83.

Integrated OCSVM mechanism for intrusion detection in SCADA systems

Leandros A. Maglaras, Jianmin Jiang and Tiago Cruz

Intrusion detection in real-time systems is a problem without a profound solution. In supervisory control and data acquisition (SCADA) systems the absence of a defence mechanism that can cope with different types of intrusions is of great importance. False positive alarms or mistakes regarding the origin of the intrusion mean severe costs for the system. An integrated one-class support vector machine (OCSVM) mechanism that is distributed in a SCADA network is presented, as a part of an intrusion detection system, providing accurate information about the origin and the time of an intrusion. The module reads the network traffic, splits traffic according to the source of the packets and creates a cluster of OCSVM models. These trained models run in parallel and can accurately and fast recognise different types of attacks.

Introduction: Cyber-physical systems are becoming vital to modernising the national critical infrastructure (CI) systems. Owing to the rapid increase of sophisticated cyber threats with exponentially destructive effects, intrusion detection systems (IDSs) must systematically evolve [1]. Most weaknesses of CIs arise from the fact that most of them are adopting on off-the-shelf technologies from the IT world, without a significant change in terms of the operator mindset, and still based on the 'airgap' security principle that suggests that an apparently isolated and obscure system is implicitly secure. Once one opens the system to off-the-shelf solutions, you also increase its exposure to cyber-attacks [2].

The design of supervisory control and data acquisition (SCADA) systems is different from conventional IT networks, even if based on the same physical technology, such as Ethernet networks. Industrial control-specific protocols are used in SCADA systems, where a limited number of packet types is exchanged between entities of the network. Moreover, the effect of some widely known attacks might have devastating consequences on SCADA systems. This situation requires the development of IDSs specifically designed for SCADA systems, and being able to ensure an adequate balance between high accuracy, low false alarm rate (FAR) and reduced network traffic overhead. Among other approaches, neural networks, support vector machines, K-nearest neighbour and hidden Markov model can be used for intrusion detection. One-class support vector machine (OCSVM) principles have shown great potential in the area of anomaly detection [3, 4].

We propose a novel mechanism, in the form of an intelligent approach based on OCSVM principles. The proposed mechanism uses a central OCSVM and a cluster of automatically produced OCSVMs, one for each source that induces significant traffic in the system, along with social metric analysis, an aggregation technique, a voting mechanism and a K-means clustering method that categorise final aggregated alerts. The mechanism is incorporated in a distributed IDS communicating with other detection and management components by producing dedicated intrusion detection message exchange format (IDMEF) messages that report the severity of the attack detected.

Proposed detection module: The main purpose of the integrated detection module is to perform anomaly detection in a time-efficient way, with good accuracy and low overhead, within a temporal window adequate for the nature of SCADA systems. To achieve the aforementioned goals, several operation stages need to be carried out:

- Selection of the most appropriate features for the training of the OCSVM models and pre-processing of raw input data.
- Creation of cluster of split OCSVM models that are trained on discrete sources. It is important to mention that we do not create a split OCSVM for each source but only for those sources that produce high traffic in the network during the training period.
- Fusion of the information gathered by the cluster of split OCSVMs and the central OCSVM in order to fire initial alarms.
- Creation of IDMEF files. The created files describe the nature of the alert, in terms of importance, the location in the system, time etc.

Features of central OCSVM: The features and their corresponding network data features are shown in Tables 1 and 2. Feature 3 examines only the packets in the last 10 packets that have the same destination host as the current packet (same host feature). Feature 4 examines only the packets in the past 10 packets that have both the same source and the destination as the current packet. For producing feature 5, each row of the dataset is assigned a value indicating the number of ARP packets in the last 10 packets.

Table 1: Features of central OCSVM model

A/A	Network date feature	Type of feature
1	Packet size	Content based
2	Rate	Time based
3	$No_{packets-dst}$	Time based
4	$No_{packets-src-dst}$	Time based
5	$No_{ARP-packets}$	Time based

Table 2: Performance evaluation of OCSVM module

Dataset	DA (%)	FAR (%)
A	98.81	1.18
B	94.6	3.25
C	95.2	1.51
D	96.37	2.3
All	96.3	2.5

Features of cluster of split OCSVMs: There are many slow probing attacks that scan the hosts using a much larger interval and merge in the overall traffic in the network. As a consequence, these attacks cannot be detected using derived time-based features. To capture these types of attacks, we modified the OCSVM module to automatically create a cluster of split OCSVMs. The features of the split OCSVM modules are a subset of the central OCSVM and include features 1, 2 and 5. Moreover, the split datasets, each consisting of the packets sent only by an individual source, adds to the detection mechanism the diversity that is crucial in order to improve the overall efficiency (dataset splitting [5]).

Social analysis: To enhance the performance of the OCSVM module, a statistical algorithm is used. For each significant source that is detected during training, a list of the protocols used is created. These protocols are ranked and stored along with the IP/MAC address of each source. During the phase testing the same procedure is executed, producing a list of the protocols used for each significant source. These lists are compared using Spearman's correlation coefficient [6]. The output of the method is a numeric value between 1 and 0, with a number closest to 1 indicating similar behaviour of the source both in training and in testing sessions. This value is used in order to give additional significance/weight to alerts produced by the corresponding source

$$p = 1 - \frac{6\sum d_i^2}{n(n^2 - 1)} \tag{1}$$

Fusion of alarms – final outputs: Each OCSVM module produces one initial alarm for any deviation in network traffic from the normal, based on its model. In real-time systems except from fast response and accuracy a limited communication between detection modules is also desirable. For this reason, outcomes of both the central OCSVM and the split OCSVMs are combined by using a simple mean value method. The values that are produced from this stage are weighted by using the social analysis algorithm. The new outcomes are then fed to the aggregation module that groups alerts per source and gives final scores to aggregated alerts by summing up values of these individual alerts. The system finally uses a K-means clustering algorithm in order to divide the alarms in three categories: possible, medium and severe. This way, the number of final alarms produced by the proposed mechanism is significantly reduced, while at the same time all sources having suspicious behaviour are reported. The proposed mechanism is represented in Fig. 1.

Nature of the trial: The trial was conducted off-line with the use of two datasets extracted from our testbed (Fig. 2). The testbed architecture mimics a small-scale SCADA system, comprising the operations and field networks and including a Human–Machine Interface Station (for process monitoring), a managed switch (with port monitoring capabilities, for network traffic capture) and two Programmable Logic

Controller Units, for process control. The NIDS and OCSVM modules are co-located on the same host, being able to intercept all the traffic flowing on the network scopes. Three kinds of attacks are evaluated:

- *Network scan attack:* In a typical network scan attack, the attacker uses TCP/FIN scan to determine if ports are closed to the target machine.
- *ARP spoofing* is a technique where an attacker sends fake ARP messages aiming to associate the attacker's MAC address with the IP address of another host. The attacker could choose to inspect the packets, modify data before forwarding ('man-in-the-middle attack (MITM)') or launch a denial of service attack by causing some of the packets to be dropped.
- *DoS flooding attack*: Network flood is the instance where the attacker floods the connection with the PLC by sending SYN packets.

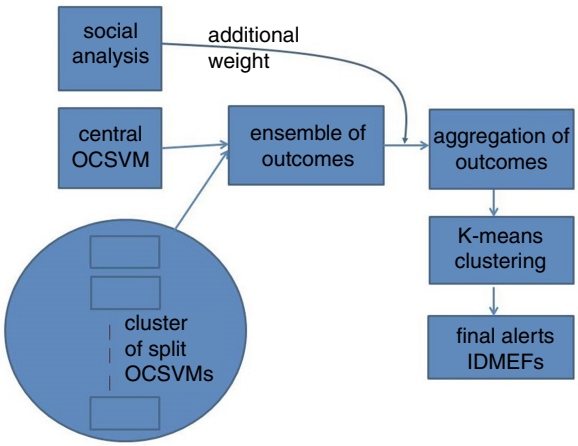

Fig. 1 *Architecture of detection mechanism*

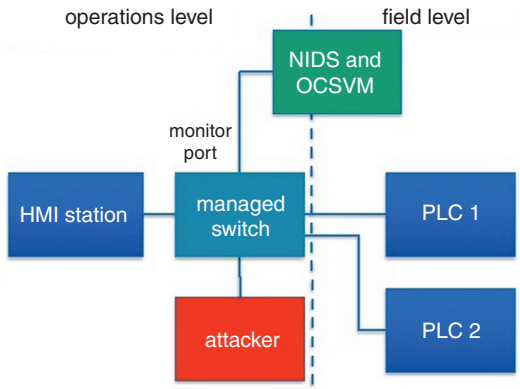

Fig. 2 *Architecture of testbed*

Results and analysis: Each OCSVM that is trained produces a separate model file which is used in order to classify new data as normal or malicious. All OCSVMs use Gaussian RBF kernel functions with parameters $\sigma = 0.01$ and $\nu = 0.001$. This Section describes the performance of the proposed OCSVM-based intrusion detection mechanism for the simulated data. The detection accuracy (DA) and FAR metrics are used as indicators to quantify the performance of the detection mechanism

$$\mathrm{DA} = \frac{\text{True positives} + \text{True negatives}}{\text{sample size}} \times 100\% \quad (2)$$

$$\mathrm{FAR} = \frac{\text{False positives}}{\text{False positives} + \text{True negatives}} \times 100\% \quad (3)$$

Since the attacks are performed during different time periods, we divide the testing dataset in several smaller ones, each containing one or at most two different attacks as follows:

Testing set-A': 1–5000: normal data: In the first dataset, there is no attack in the system and the model produces 60 alarms and 16 aggregated final alarms (IDMEF files). These alarms are categorised as severe (6) alarm and possible (10).
Testing set-B': 5000–10 000: normal data + ARP spoofing attack + network scan: In the second dataset, the system fires 658 initial alarms and 21 aggregated IDMEF files. The majority of the alerts are concerning two sources. The first is 127.27.1.99 which is used in order to capture the traffic and produce the network trace. This capturing of the network traffic is translated from our module as an attack and an alarm is fired. The second is 00:0C:29:9a:5f:7a which is the MAC address of the intruder. The final aggregated output of the OCSVM module is two severe alarms (for the actual attackers) and 19 possible attacks for packets that are classified as abnormal but with low risk.
Testing set-C': 10 000–25 000: normal data + DoS flooding attack + network scan: In the third dataset, the system fires 9273 initial alarms and 18 final aggregated IDMEF files. One severe alarm is fired for source 172.27.224.33 which is the IP address of the intruder. This alarm is classified as severe.
Testing set-D': 25 000–41 000: normal data + MITM attack: In the fourth dataset, the system triggers 203 initial and 16 final alarms from the classification algorithm (2 severe and 14 possible alarms). The MITM attack that is conducted by source 00:0C:29:9a:5f:7a is not classified as an attack due to low activity in this time period but the intruder was identified earlier when it was conducting an ARP spoofing attack, which is the preliminary step before the MITM attack.
Full testing dataset – combined attacks: When testing the OCSVM module with the full dataset which contains all the attacks, the system produces 10 507 initial alarms and 22 final aggregated alarms. The final alarms contain three severe alarms for the sources 127.27.1.99 (traffic capture machine), 00:0C:29:9a:5f:7a (MAC address of the intruder) and 172.27.224.33 (IP address of the intruder) and 19 possible attacks. The overall performance of the system is good and all the real attacks are classified as severe alerts.

Conclusion: This Letter proposes a new integrated mechanism for intrusion detection in SCADA systems. The mechanism uses network traces in order to automatically create a cluster of split OCSVMs. Outcomes of split OCSVMs are combined with those of the central OCSVM, are weighted using social metrics, aggregated and finally classified using K-means clustering. Final outputs of the mechanism are communicated with the use of IDMEF messages each one reporting the severity of the detected attack. The proposed mechanism produces a small amount of final alerts and manages to detect all the attacks efficiently. As future work, the proposed mechanism will be evaluated against other attack scenarios and will be enhanced in order to improve FAR.
Acknowledgments: This research is supported by the project 'CockpitCI: Cybersecurity on SCADA: risk prediction, analysis and reaction tools for Critical Infrastructures', funded under the European Framework-7 Programme (contract no. 285647).

14 August 2014
doi: 10.1049/el.2014.2897
One or more of the Figures in this Letter are available in colour online.

Leandros A. Maglaras and Jianmin Jiang (*Department of Computing, Faculty of Engineering and Physical Sciences, University of Surrey, Guildford, GU2 7XH, United Kingdom*)

E-mail: l.maglaras@surrey.ac.uk

Tiago Cruz (*Department of Informatics Engineering, University of Coimbra, Coimbra 15780, Portugal*)

References

1 Al-Sakib Khan, P.: 'The state of the art in intrusion prevention and detection' (CRC Press, London, UK, 2014)
2 Cruz, T., Jorge, P., Paulo, S., Matthieu, A., Moussa, O., Antonio, G., and Lasith, Y.: 'Improving cyber-security awareness on industrial control systems: the CockpitCI approach'. Proc. of 13th ECCWS 2014 Piraeus, Greece, July 2014
3 Wang, Y., Wong, J., and Miner, A.: 'Anomaly intrusion detection using one class SVM'. Proc. 5th Annual IEEE SMC on Information Assurance Workshop, West Point, NY, USA, June 2004, pp. 358–364
4 Maglaras, L.A., and Jiang, J.: 'Intrusion detection in SCADA systems using machine learning techniques'. Proc. of SAI Conf., London, UK, August 2014
5 Krawczyk, B., and Wozniak, M.: 'Diversity measures for one-class classifier ensembles', *Neurocomputing*, 2014, **126**, pp. 36–44
6 Spearman, C.: 'The proof and measurement of association between two things', *Am. J. Psychol.*, 1904, **15**, (1), pp. 72–101

OCSVM model combined with K-means recursive clustering for intrusion detection in SCADA systems

Leandros A. Maglaras, Jianmin Jiang
Department of Computing , University of Surrey, Guildford, UK
{l.maglaras, jianmin.jiang}@surrey.ac.uk

***Abstract*—Intrusion detection in Supervisory Control and Data Acquisition (SCADA) systems is of major importance nowadays. Most of the systems are designed without cyber security in mind, since interconnection with other systems through unsafe channels, is becoming the rule during last years. The de-isolation of SCADA systems make them vulnerable to attacks, disrupting its correct functioning and tampering with its normal operation.**

In this paper we present a intrusion detection module capable of detecting malicious network traffic in a SCADA (Supervisory Control and Data Acquisition) system, based on the combination of One-Class Support Vector Machine (OCSVM) with RBF kernel and recursive k-means clustering. The combination of OCSVM with recursive k-means clustering leads the proposed intrusion detection module to distinguish real alarms from possible attacks regardless of the values of parameters σ and ν, making it ideal for real-time intrusion detection mechanisms for SCADA systems. The OCSVM module developed is trained by network traces off line and detect anomalies in the system real time. The module is part of an IDS (Intrusion Detection System) system developed under CockpitCI project.

I. Introduction- Motivation

Several techniques and algorithms have been reported by researchers for intrusion detection. One big family of intrusion detection algorithms is rule based algorithms. In real applications though, during abnormal situations, the behavior of the system cannot be predicted and does not follow any known pattern or rule. This characteristic makes rule based algorithms incapable of detecting the intrusion.

An intelligent approach based on OCSVM [One-Class Support Vector Machine] principles are proposed for intrusion detection. OCSVM is a natural extension of the support vector algorithm to the case of unlabeled data, especially for detection of outliers. The OCSVM algorithm maps input data into a high dimensional feature space (via a kernel) and iteratively finds the maximal margin hyperplane which best separates the training data from the origin. OCSVM principles have shown great potential in the area of anomaly detection [1], [2]. Several extensions of OCSVM method have been introduced lately [3], [4].

For the OCSVM with an RBF kernel, two parameters σ and ν need to be carefully selected in order to obtain the optimal classification result. A common strategy is to separate the data set into two parts, of which one is considered unknown. The prediction accuracy obtained from the unknown set more precisely reflects the performance on classifying an independent data set. An improved version of this procedure is known as cross-validation. Cross-validation is a model validation technique for assessing how the results of a statistical analysis will generalize to an independent data set. It is mainly used in settings where the goal is prediction, and one wants to estimate how accurately a predictive model will perform in practice.

Unnthorsson et al. [5] proposed a method to select parameters for the OCSVM. In their method, ν was first set to a user-specified allowable fraction of misclassification of the target class (e.g. 1% or 5%), then the appropriate σ value was selected as the value for the classification accuracy curve of training samples first reaches $1-\nu$. The obtained ν and σ combination can then be used in the OCSVM classification.

OCSVM similar to other one-class classifiers suffer from false positive and over fitting. The former is a situation that occurs when the classifier fires an alarm in the absence of real anomaly in the system and happens when parameter σ has too large vale. The latter is the situation when a model begins to memorize training data rather than learning to generalize from trend and it shows up when parameter σ is given relatively small value [6]. In this article we propose the combination of OCSVM method with a recursive k-means clustering, separating the real from false alarms in real time and with no pre-selection of parameters σ and ν.

II. $\mathcal{K}-\mathcal{OCSVM}$

The proposed $\mathcal{K}-\mathcal{OCSVM}$ combines the well known OCSVM classifier with the RBF kernel with a recursive K-means clustering module. Figure 1 illustrates the procedure of intrusion detection of our proposed $\mathcal{K}-\mathcal{OCSVM}$ model.

The OCSVM classifier runs with default parameters and the outcome consists of all possible outliers. These outliers are clustered using the k-means clustering method with 2 clusters, where the initial means of the clusters are the maximum and the minimum negative values returned by the OCSVM module. From the two clusters that are created from the K-means clustering, the one that is closer to the maximum negative value (severe alerts) is used as input in the next call of the K-means clustering. This procedure is repeated until all outcomes are put in the same cluster or the divided set is big enough compared to the initial one, according to the threshold parameter k_{thres}.

K-means clustering method divides the outcomes according to their values and those outcomes with most negative values are kept. That way, after the completion of this recursive procedure only the most severe alerts are communicated from

Research supported by the project "CockpitCI: Cybersecurity on SCADA: risk prediction, analysis and reaction tools for Critical Infrastructures ", funded under European Framework-7 Programme (contract No. 285647)

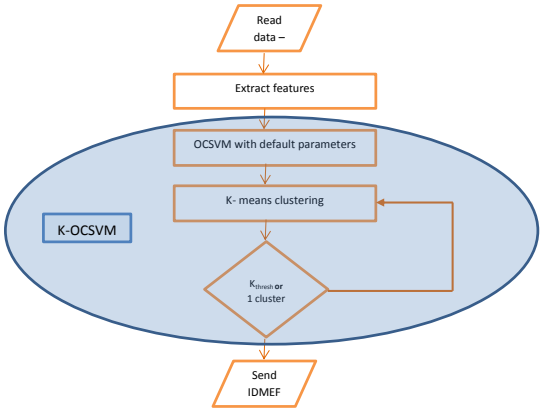

Fig. 1: $\mathcal{K}-\mathcal{OCSVM}$ module

the $\mathcal{K}-\mathcal{OCSVM}$. The division of the data need no previous knowledge about the values of the outcomes which may vary from -0.1 to -160 depending of the assigned values to parameters σ and ν. The method can find the most important/possible outliers for any given values to parameters σ and ν.

One important parameter that affects the performance of $\mathcal{K}-\mathcal{OCSVM}$ is the value of threshold k_{thres}. For given value 2, the final cluster of severe alerts that the method communicates to other parts of the IDS system is limited to 2 to 4 alarms. For bigger value (3 or more) the number of alerts also rises till the method degrades to the initial OCSVM. The optimal value for the given parameter k_{thres} is a matter for future investigation.

III. Performance evaluation

A. Training of OCSVM model

Training of OCSVM module was conducted using a trace file that is sniffed out of a typical wireless network that consists of 10.000 lines each representing a packet send in the network. To train the OCSVM, we adopt the RBF for the kernel equation. This kernel nonlinearly maps samples into a higher dimensional space so it can handle the case when the relation between class labels and attributes is nonlinear.

The training model that is extracted after the training of the OCSVM is used for on line detection of malicious data. Since the model is based on features that are related to network traffic, and since the traffic of the system varies from area to area and from time period to time period, possible generation of multiple models could improve the performance of the module.

B. Testing of OCSVM model

In order to test our model we use another network trace files sniffed from the wireless network. The testing trace file consists of 30.000 lines. We compare the performance of our proposed model against OCSVM classifiers having the same values for parameters σ and ν.

The parameters used for the evaluation of the performance of $\mathcal{K}-\mathcal{OCSVM}$ are listed in Table I.

Parameter	*Range of Values*	*Default value*
σ	0.1 - 0.0001	0.007
ν	0.002 - 0.05	0.01
$Threshold$	2-3	2

TABLE I: Evaluation Parameters

In Table II we show the number of observed anomalies detected from OCSVM and $\mathcal{K}-\mathcal{OCSVM}$ respectively. From this table it is shown how parameters σ, ν affect the performance of OCSVM. Even for a value of ν equal to 0.005, OCSVM produces almost 500 possible attacks, making the method inappropriate for a SCADA system where each false alarm is costly.

Parameter σ	*Parameter* ν	$\mathcal{K}-\mathcal{OCSVM}$	*ocsvm*
0.007	0.002	3	408
0.007	0.01	3	299
0.007	0.005	2	408
0.0001	0.01	3	274
0.1	0.01	2	295

TABLE II: Performance evaluation of $\mathcal{K}-\mathcal{OCSVM}$ and OCSVM for $K_{thers} = 2$.

IV. Conclusions

We have presented a intrusion detection module for SCADA systems that is based in OCSVM classifier and a recursive k-means clustering method. The module is trained off-line by network traces, after the attributes are extracted from the network dataset. The intrusion detection module is part of an IDS system developed under CoCkpitCI.

After the completion of the $\mathcal{K}-\mathcal{OCSVM}$ method only severe alerts are communicated to the system by IDMEF files that contain information about the source, destination, protocol and time of the intrusion. The method is stable and its performance is not influenced by the selection of parameters ν and σ. Further analysis is needed in order to evaluate the performance of the method under different attack scenarios.

References

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

[2] L. Maglaras and J. Jiang, "Intrusion detection in scada systems using machine learning techniques," in *Proceedings of the 2nd SAI conference*. SAI, 2014.

[3] A. Glazer, L. Michael, and S. Markovitch, "q-ocsvm: A q-quantile estimator for high-dimensional distributions," in *In Proceedings of The 27th Conference on Neural Information Processing Systems (NIPS-2013),Lake Tahoe, Nevada*, 2013.

[4] X. Song, G. Fan, and M. Rao, "Svm-based data editing for enhanced one-class classification of remotely sensed imagery," *Geoscience and Remote Sensing Letters, IEEE*, vol. 5, no. 2, pp. 189–193, 2008.

[5] T. P. Runarsson and M. T. Jonsson, "Model selection in one-class ν-svms using rbf kernels," in *Proceedings of 16th International Congress and Exhibition on Condition Monitoring and Diagnostic Engineering Management*, 2003.

[6] X. Li, L. Wang, and E. Sung, "Adaboost with svm-based component classifiers," *Engineering Applications of Artificial Intelligence*, vol. 21, no. 5, pp. 785–795, 2008.