OSCIDS: An Ontology based SCADA Intrusion Detection Framework

Abdullah Al Balushi, Kieran McLaughlin and Sakir Sezer
Centre for Secure Information Technologies,
Queens University Belfast, Belfast BT3 9DT, U. K.

Keywords: Semantic Web, Intrusion Detection, Knowledge Engineering, SCADA, Modbus TCP, Security Ontology.

Abstract: This paper presents the design, development, and validation of an ontology based SCADA intrusion detection system. The proposed system analyses SCADA network communications and can derive additional information based on the background knowledge and ontology models to enhance the intrusion detection data. The developed intrusion model captures network communications, cyber attacks and the context within the SCADA domain. Moreover, a set of semantic rules were constructed to detect various attacks and extract logical relationships among these attacks. The presented framework was extensively evaluated and a comparison to the state of the art is provided.

1 INTRODUCTION

Supervisory Control and Data Acquisition (SCADA) systems are used to monitor and control various critical infrastructures such as electrical generation and distribution, petroleum plants, or nuclear plants. In the past, SCADA networks operated in closed environments using legacy communication protocols. However, modern networks are moving towards the use of open standards and general information and communications technology (ICT) to enhance their systems interoperability, reduce cost, and enhance operational management. For instance, the Modbus serial communication protocol developed by Modicon in 1979 is now being encapsulated into TCP/IP frames in the newer version of Modbus TCP/IP, also referred as Modbus TCP (Modbus, 2012).

Nevertheless, the incorporation of such open technologies in SCADA networks exposed them to a wide range of vulnerabilities and cyber-attacks such as reconnaissance, command injection, denial of service, and response and measurement injection. As a result, the number of cyber incidents is significantly increasing (Kang et al., 2014) which presents novel challenges for both academic researchers and industry experts.

In order to provide an adequate level of protection to SCADA networks, various types of SCADA-specific intrusion detection systems (IDSs) have been proposed (Zhu and Sastry, 2010). However, the current generation of IDS systems in SCADA networks are often faced with several challenges. First, the majority of the existing systems, especially signature-based, are purely syntactic and lack a clear description of intrusion behaviours and semantics of the monitored systems (Barry and Chan, 2009). This issue may result in missing the detection of sophisticated attacks (Hadžiosmanović et al., 2014) such as Stuxnet (Langner, 2011) that may exploit legitimate looking commands to cause damage on the system. Second, intelligent capabilities such as reasoning over existing domain knowledge, which can assist in identifying the logical relationships between various attacks are missed by these IDS systems. This issue can result in the need for intensive expertise and efforts to manually analyse the connection between these attacks.

To address the above limitations, this paper presents an ontology-based intrusion detection system called OSCIDS (**O**ntology-based **SC**ADA **I**ntrusion **D**etection **S**ystem). OSCIDS is a knowledge-based system that utilises ontology to extract semantic relations among attacks and detect intrusions. The developed ontology enables representing the main concepts and semantic relations of network communications, cyber-attacks, and inferring additional useful knowledge from the existing background knowledge. **The main contributions** of this paper are:

- A novel approach for intrusion detection in SCADA networks using ontologies has been proposed. In contrast to current signature-based techniques, the proposed framework utilises the semantic definitions of industrial network packets

Balushi, A., McLaughlin, K. and Sezer, S.
OSCIDS: An Ontology based SCADA Intrusion Detection Framework.
DOI: 10.5220/0005969803270335
In *Proceedings of the 13th International Joint Conference on e-Business and Telecommunications (ICETE 2016) - Volume 4: SECRYPT*, pages 327-335
ISBN: 978-989-758-196-0

and logical relationships between attacks to enhance the intrusion detection process.

- Formal semantic modelling of SCADA cyber intrusions and specification of Modbus TCP/IP industrial communications is presented. The constructed model provides an expressive representation of possible intrusions and enables new intrusion analysis capabilities such as the use of reasoning over existing knowledge and deriving additional information from the raw packets.
- A set of semantic rules for background knowledge inferences and intrusion detection has been developed. These rules can utilise both raw packet features and the derived features by the reasoning process to detect various attacks or find their logical relationship.
- The proposed framework prototype is developed using open-source technologies. This enables system extensibility and flexibility. Moreover, an evaluation of the proposed framework is provided with a comparison to the state-of-the-art IDS solution.

This paper is organised as follows. Section 2 presents the background and related works. Section 3 discusses the development and the architecture of proposed OSCIDS framework. Section 4 presents the experimental results, while Section 5 concludes the paper.

2 BACKGROUND AND RELATED WORK

2.1 Modbus TCP/IP Communications Overview

Modbus is a serial communications protocol developed by Modicon (now Schneider Electric) in 1979 for use in industrial networks. Modbus TCP, on the other hand, refers to the encapsulated Modbus communications into TCP/IP frames. The Modbus TCP has a simple request/response architecture. A Modbus client (Master) sends a packet to the Modbus server requesting a specific command to be executed or to return some data. Various commands can be requested which may include reading process measurements, writing configurations, performing diagnostics operations, or executing a process control operation. The server processes the received command and in many cases returns a response indicating the status of command execution or describing any errors which occurred.

Modbus TCP packet contains several features at both network and application layers. The main features of Modbus TCP packets are illustrated in Figure 1. These features include the TCP/IP headers and the Modbus Application Layer (known as Modbus Application Data Unit(ADU)) features.

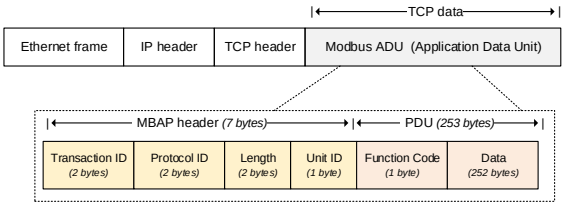

Figure 1: Modbus TCP packet structure.

2.2 SCADA-specific Intrusion Detection Systems

In the literature, there exist a number of IDS proposals that are specifically addressing the intrusion detection problem in SCADA networks (Zhu and Sastry, 2010). We will present some of the recent IDS proposals with reference to Modbus communications.

Snort (Roesch et al., 1999) is an open source signature-based intrusion detection system used in many networks. Digital Bond Inc.(Peterson, 2009) has developed a Modbus pre-processor and a set of 14 attack signature rules that captures various attacks. These attacks are mainly reconnaissance, denial of service, and unauthorised access commands. (Morris et al., 2013) presented a set of 50 Snort rules for attack detection on Modbus communications over serial lines and TCP/IP.

Other approaches that rely on anomaly detection techniques have been proposed. (Mallouhi et al., 2011) proposed an anomaly-based IDS for industrial protocols including Modbus TCP/IP. Their system consists of a Modbus and TCP analysers that use the n-gram algorithm for intrusion detection. This system was developed to detect several attacks such as spoofing, TCP SYN flood, Modbus-specific attacks, Denial of service, and Man in the middle attacks. Nevertheless, the authors provided only preliminary results. (Carcano et al., 2011) proposed an intrusion detection system for Modbus protocol in power plants. Their approach is based on the concept of Critical State Analysis and State Proximity. (Sayegh et al., 2014) presented an anomaly-based intrusion detection system developed for SCADA networks. In their approach, time correlation between different packets is extracted to identify malicious events.

Besides the above IDS techniques, an emerging research area for the utilisation of Artificial Intelligence (AI) techniques such as ontology knowledge

representation, has been approached by researchers and industry experts to design, implement, and enhance intrusion detection systems.

2.3 Ontology-based Approaches for Intrusion Detection

An ontology formally represents the knowledge within a specific domain through a list of concepts and their relationship in a format that is easy to understand by human and machines. It provides machines with powerful semantic-level capabilities such as logical inferences and reasoning over existing knowledge. However, the use of ontology approaches for security monitoring in SCADA domain is still in its infancy and little work is published in this regard.

(Barnett et al., 2012) developed an ontology for the security of Smart grids. The proposed ontology focuses on both physical and network specific risks. The authors also developed a graphical user interface (GUI) that displays the results of a Failures mode effects analysis based on the prioritisation by likelihood, detectability, and severity. However, the built ontology is simple and may not be suitable for intrusion detection. (Sheldon et al., 2013) proposed an ontology to assess root causes and impacts of cyber intrusions on SCADA systems. Nevertheless, the evaluation results are not provided in this work. (Sartakov, 2015) proposed the use of an ontological representation of networks for intrusion detection in Cyber-Physical systems. The proposed approach combines both logical and physical elements of the network that are used to build a specification model. Nevertheless, the proposed ontology is very general and it is not clear whether some specific industrial protocols are considered in the modelling or not.

The use of ontology has also appeared in several European Framework Programmes (FP7) projects and practical implementations, but with different objectives than for intrusion detection. In SERSCRIS project (Surridge et al., 2012), an ontology was constructed to describe generic system assets, threats, security controls and their relationships in critical infrastructure networks. However, the scope of published work is limited to the modelling of threats and vulnerabilities rather than using it for run-time monitoring. In another project named INSPIRE (Choraś et al., 2010), a decision-aid tool for diagnostic and recovery is proposed using ontology. The constructed ontology describes SCADA interdependencies, vulnerabilities, and threats. However, the proposed ontology is focused on diagnostics and may not be suitable for our intrusion detection objective.

Despite the recent advances in security researches for industrial control systems (ICS), it is clear that the use of ontology approaches for intrusion detection in SCADA domain is limited. An ontology can provide several advantages and enables enhanced information analysis to the intrusion detection task. This includes representing the knowledge in a clear structure and with powerful semantic-level capabilities such logical inferences and reasoning over existing knowledge. Therefore, we propose an ontology-based intrusion detection system for use in SCADA networks.

3 ONTOLOGY-BASED SCADA IDS (OSCIDS)

The ontology-based intrusion detection framework presented in this paper is a knowledge-based system. It utilises the power of semantic data definitions to capture and represent intrusion-related knowledge within the domain in a formal language that is understood by both humans and machines. This data representation, backed up by the semantic definitions, enables expressing the relationships among the hierarchically organised concepts of the data. Therefore, new capabilities related to the classification and reasoning over the existing data are made possible. The OSCIDS framework is prototyped in Java language on top of the open-source Apache Jena API framework (Jena, 2011). Jena provides an extensive set of features for processing ontology such as triples storage, inferences and rule engines, and querying engines.

3.1 Architecture

The overall architecture of OSCIDS is presented in Figure 2. It consists of several components that are described as follows.

- **Knowledge Base (KB):** This component provides a repository storage for all the knowledge elements used by the framework. This includes the ontological models, processed network packets, attack instances, classes relationships, semantic rules, and any other related information. This component is implemented using JENA TDB (Triple Store Database) which provides a persistent storage on hard drives with a high performance and the datasets are protected against accidental corruption, unexpected process terminations and system crashes.
- **Semantic Protocol Analyser:** This component is responsible for the processing of raw network

packets and the extraction of main features including the Modbus application-layer as per the developed ontology model. Then, it creates a packet instance in the knowledge base under a proper class such as Modbus TCP instance. This component is developed in Java language using JNETPCAP library with integration to Jena API functions to allow accessing and modifying the data in the KB.

- **Inference and Rule Engines:** This component of the framework enables rules evaluation on the existing knowledge (such as packet instances) and the retrieval of information stored in the KB. Using a set of ontological statements (such as restrictions and truth conditions), it can automatically derive additional information to enhance the quality of existing data. For example, new semantic features such as the packet type, the application layer payload type, the characteristic of the used industrial command can be generated out of the raw packet features.
- **Classification and Detection Module:** This module facilitates the communication interface between all the system components and controls the overall workflow. It is responsible for guiding other components to the intrusion detection task and taking the final decisions when required.

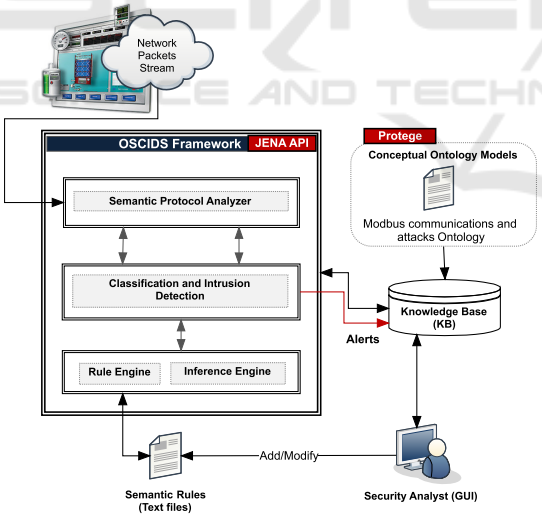

Figure 2: Overall OSCIDS Framework Architecture.

3.2 Workflow

The main activities of the framework are illustrated in figure 3, and the step by step operating process is as follows:

The raw network packets are captured and fed into the system and handled by the semantic protocol analyser (*Step 1*). The semantic protocol analyser retrieves the network communications model (ontology) (*Step 2*) and uses the encoded knowledge about Modbus TCP/IP and other defined communications characteristics to extract the main packet features. The resultant packet with its extracted features is converted to a Resource Description Framework (RDF) format and added to the knowledge base under a proper class as Modbus-TCP, TCP (non-Modbus), or UDP packet instances (*Step 3*). Next, the process handling task is forwarded to the classification and detection module (*Step 4*). This module instructs the inference and rule engine to execute various semantic rules for reasoning and attack detection (*Step 5*). These rules are loaded from text files (*Step 6*) and their execution results are returned back (*Step 7*). If the system determined that a malicious packet has been found, an alert is created in the KB and sent to the security analyst (*Step 8*).

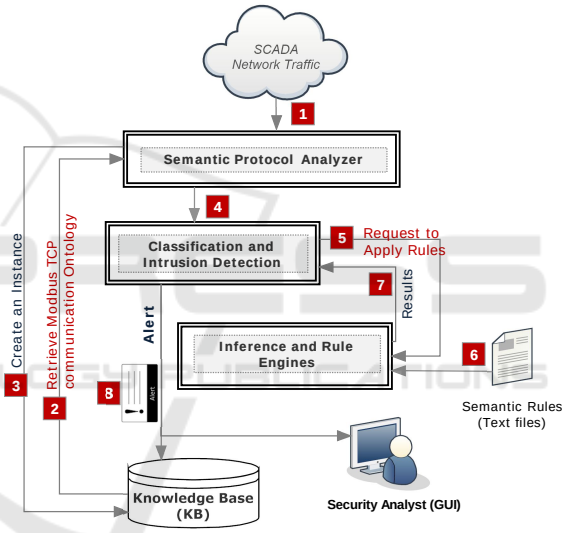

Figure 3: OSCIDS Framework Work-flow.

3.3 Ontology Development

The SCADA intrusion knowledge is captured and encoded using ontology. The constructed ontology defines the Modbus TCP communications, Cyber-attacks, and the logical relationships among packet and attack instances. This model is derived by analysing the protocol specifications (Modbus, 2012) and attack taxonomies (Drias et al., 2015; Hadžiosmanović et al., 2014). A high-level representation of the developed ontology is depicted in Figure 4.

The main concepts (classes) contained in this ontology are presented in Table 1. The described classes, which include TCP packet, Modbus TCP/IP Packet, and Attack instance, are used to describe different types of data and their definition. Each of these

classes may have several subclasses and properties organised in a hierarchy. For instance, the Modbus TCP class has several subclasses such as Request-Message, Response-Message, and Modbus-Header.

Table 1: Main classes contained in the SCADA intrusion ontology.

Class	Description
TCP Packet	This class represents an instance of a network packet. This includes all TCP packets whether it has Modbus communication or not.
Modbus Packet	This class is used to represent instances of Modbus TCP packets and their features.
Attack	This class represents the cyber-attack instances on SCADA systems. It has several attack information such as impact, affected systems, attack description, and attack vector.

Moreover, in ontology, the relationship between classes are defined using object properties, while the data properties are used to describe the data values, as presented in Table 2. The logical relationships describe the connection between different elements of an attack. For example, an **[Attacker]** performs an **[Attack]** which occurred at a specific **[Time]** and has an **[Impact]** on the **[Target]**. This **[Attack]** uses a **[Mechanism]** to exploit a **[Vulnerability]** utilising a **[Protocol]** packet message.

Table 2: Properties contained in the SCADA intrusion ontology.

Domain	Properties
TCP packet	hasTime, hasTCPLength, hasSrcIP, hasSrcPort, hasSrcMAC, hasDstIP, hasDstPort, hasDstMAC
Modbus packet	hasModbusTransID, hasModbusUnitID, hasModbusProtoID, hasModbusLen, hasModbusFC, hasModbusSubFC, hasMExcepCode, hasModbusPacketType
Network packet	hasImpact, hasCount, isMalicious, hasCorrelationID, hasAttackVector

3.4 Semantic Rules Generation

In this paper, a set of 15 rules were developed to validate the proposed system capabilities in detecting malicious Modbus communications, as shown in Listing 1. These rules cover various attacks such as denial of service, protocol specification violation, system integrity, and reconnaissance attacks. Some rules were also used to derive additional knowledge from the raw packet data that can be used by other attack detection rules.

Listing 1: Semantic detection rules using Jena format.

```
[Rule01: (?p ics:hasSrcPort 502) −> (?p ics:hasPacketType "Response")]
[Rule02: (?p ics:hasDstPort 502) −> (?p ics:hasPacketType "Request")]
[Rule03: (?p ics:hasTCPlen ?TCPLen), greaterThan(?TCPLen,300) −> (?p ics:hasImpact "Specification−Violation")(?p ics:hasAttackVector "Oversized packet length")(?p ics:isMalicious "Yes")]
[Rule04: (?p ics:hasProtoID ?ProtocolID), notEqual(?ProtocolID,0) −> (?p ics:hasImpact "Specification−Violation")(?p ics:hasAttackVector "Illegal ModbusTCP Protocol ID")(?p ics:isMalicious "Yes")]
[Rule05: (?p ics:hasModbusLen ?Len), greaterThan(?Len,254) −> (?p ics:hasImpact "Specification−Violation")(?p ics:hasAttackVector "Oversized Modbus packet length")(?p ics:isMalicious "Yes")]
[Rule06: (?p ics:hasUnitID ?UnitID), greaterThan(?UnitID,247) −> (?p ics:hasImpact "Specification−Violation")(?p ics:hasAttackVector "Illegal Modbus Unit ID")(?p ics:isMalicious "Yes")]
[Rule07: (?p ics:hasPacketType "Request"),(?p ics:hasFunctionCode ?FC),greaterThan(?FC,127) −> (?p ics:hasImpact "Specification−Violation")(?p ics:hasAttackVector "Out of range Modbus Function Code")(?p ics:isMalicious "Yes")]
[Rule08: (?p ics:hasFunctionCode 8)(?p ics:hasSubFunctionCode 10) −> (?p ics:hasImpact "System−Integrity")(?p ics:hasAttackVector "CLEAR−AUDIT−COUNTERS−DIAG−REGISTERS")(?p ics:isMalicious "Yes")]
[Rule09: (?p ics:hasFunctionCode 8)(?p ics:hasSubFunctionCode 4) −> (?p ics:hasImpact "Denial−of−Service")(?p ics:hasAttackVector "FORCE−LISTEN−ONLY−MODE")(?p ics:isMalicious "Yes")]
[Rule10: (?p ics:hasFunctionCode 8)(?p ics:hasSubFunctionCode 1) −> (?p ics:hasImpact "Denial−of−Service")(?p ics:hasAttackVector "RESTART−COMMUNICATIONS−OPTION")(?p ics:isMalicious "Yes")]
[Rule11: (?p ics:hasFunctionCode 43) −> (?p ics:hasImpact "System−Reconnaissance")(?p ics:hasAttackVector "READ−DEVICE−ID")(?p ics:isMalicious "Yes")]
[Rule12: (?p ics:hasExceptionCode 6) −> (?p ics:hasImpact "Denial−of−Service")(?p ics:hasAttackVector "SLAVE−DEVICE−BUSY")(?p ics:isMalicious "Yes")]
[Rule13: (?p ics:hasFunctionCode 17) −> (?p ics:hasImpact "System−Reconnaissance")(?p ics:hasAttackVector "REPORT−SERVER−INFORMATION")(?p ics:isMalicious "Yes")]
[Rule14: (?p ics:hasExceptionCode 5) −> (?p ics:hasImpact "Denial−of−Service")(?p ics:hasAttackVector "ACKNOWLEDGE−EXCEPTION−CODE−DELAY")(?p ics:isMalicious "Yes")]
[Rule15: (?p ics:hasDstPort 502)(?p ics:hasPayloadType "Non−Modbus−TCP") −> (?p ics:hasImpact "Specification−Violation")(?p ics:hasAttackVector "Non−Modbus Communication on TCP Port 502")(?p ics:isMalicious "Yes")]
```

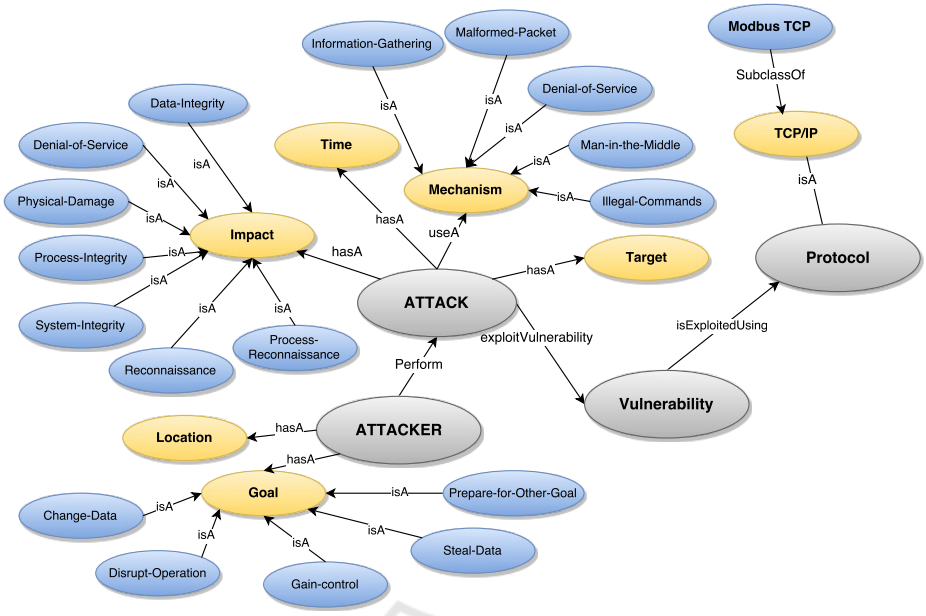

Figure 4: High-level view of the proposed SCADA intrusion ontology.

4 OSCIDS FRAMEWORK EVALUATION

In order to validate the proposed OSCIDS framework, we conducted several experiments using multiple datasets. The experimental results and comparison to state of the art solutions are provided in this section. Snort IDS with the QuickDraw (Peterson, 2009) Modbus preprocessor and rule-set is used for comparison as a state-of-the-art solution.

4.1 Experiments Setup and Datasets

The architecture of research test-bed used for the experiments is illustrated in Figure 5. It contains several components including 1) A simulated Modbus Server using ConPot industrial honey-pot. 2) A Modbus Client with a Python script that is configured to poll Modbus measurements at a defined time interval for the generation of benign traffic. 3) An attacker machine with KALI-Linux distribution for penetration testing. 4) Snort IDS with Quick-draw. 5) The proposed OSCIDS framework prototype.

Furthermore, two datasets of network traffic captures (PCAP) were used. The first set was obtained from Digital Bond (Peterson, 2009). The second dataset is constructed using our research test-bed components, which includes both benign and malicious traffic samples.

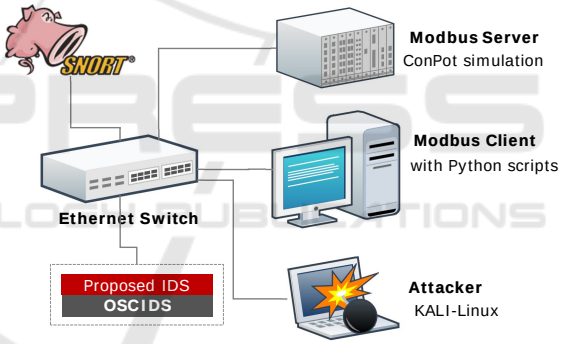

Figure 5: Experimental Test-bed Architecture.

4.2 Experiment 1: Ontology-based vs. Snort Intrusion Detection

This experiment aims at evaluating the attack detection capability of our proposed system and providing a comparison to state-of-the-art Snort IDS with QuickDraw (Peterson, 2009). The experimental process was developed as follows. First, the two datasets described previously are fed into running instances of Snort IDS (*with QuickDraw*) and the proposed IDS framework. The results of each system were recorded and used for comparison.

In this experiment, a total of 468 network packets were processed by our system. Out of these packets, 331 Modbus-TCP packets were converted to RDF instances with the class type of Modbus Packet, while 137 (Non-Modbus) packets were added as TCP

Packet instances.

The proposed system was successfully able to derive a set of additional features to the raw packet information. These features include, but not limited to, the packet payload type (TCP or Modbus TCP), Modbus message type (Request/Response), function or exception codes description. Furthermore, the evaluation of semantic detection rules on the packets resulted in identifying malicious packet instances. The detected attack includes System-Reconnaissance (51.14%), Denial-of-Service (12.7%), Specification-Violation (0.09%), System-Integrity (0.18%), and Miscellaneous (33%). An example Modbus packet instance with the extended features in RDF triples format is illustrated in Listing 2.

Listing 2: A malicious Modbus packet instance in RDF format.

```
<rdf:Description rdf:about="http://
    SCADA.ICS/OIDS/networktraffic.OWL#
    MODBUSTCPpacket22">
 <ics:isMalicious>Yes</ics:isMalicious>
 <ics:hasTime>2015-08-22 13:01:44
     .837161</ics:hasTime>
 <ics:hasAttackVector>CLEAR-AUDIT-
     COUNTERS-DIAG-REGISTERS</
     ics:hasAttackVector>
 <ics:hasImpact>System-Integrity</
     ics:hasImpact>
 <ics:hasPacketType>Request</
     ics:hasPacketType>
 <ics:hasUnitID>10</ics:hasUnitID>
 <ics:hasModbusLen>6</ics:hasModbusLen>
 <ics:hasTransID>0</ics:hasTransID>
 <ics:hasDstPort>502</ics:hasDstPort>
 <ics:hasPayloadType>Modbus-TCP</
     ics:hasPayloadType>
 <ics:hasFunctionCodeLabel>Diagnostics
     (08)</ics:hasFunctionCodeLabel>
 <ics:hasSrcMAC>00:20:78:00:62:0D</
     ics:hasSrcMAC>
 <ics:hasProtoID>0</ics:hasProtoID>
 <ics:hasTCPlen>66</ics:hasTCPlen>
 <ics:hasDstMAC>00:02:B3:CE:70:51</
     ics:hasDstMAC>
 <ics:hasDstIP>192.168.0.3</
     ics:hasDstIP>
 <ics:hasFunctionCode>8</
     ics:hasFunctionCode>
 <ics:hasSrcPort>2578</ics:hasSrcPort>
 <ics:hasSrcIP>192.168.0.57</
     ics:hasSrcIP>
 <ics:hasSubFunctionCode>10</
     ics:hasSubFunctionCode>
</rdf:Description>
```

Moreover, a brief security alert is forwarded to the security analyst or console in standard Syslog format (RFC 3164) as following:

```
<0> 2015-08-22 13:01:44.837161 OSCIDS IDS SB-011
    Suspicious Modbus Diagnostic code (CLEAR-
    AUDIT-COUNTERS-DIAG-REGISTERS) has been
    successfully executed on a Modbus device
    192.168.1.10 2578 192.168.1.222 502
```

A comparison of the attack detection rate, false positives, and accuracy between the proposed system and Snort IDS is presented in Table 3. The proposed system was also assessed on its performance. It took the system 492.77 Milliseconds *(0.49277 Second)* to process and create 468 packet instances in RDF format. While applying the 15 semantic rules for inference and attack detection took 21.86 Milliseconds *(0.02186 Second)*.

Table 3: Comparison of attack detection rates between the proposed OSCIDS and Snort IDS.

IDS	Detection Rate (%)	False Positive (%)	Accuracy (%)
OSCIDS	97.80	1.09	96.70
Snort IDS	96.70	1.64	94.50

4.3 Experiment 2: Complex Attack Detection and Correlation

In this experiment, we assess the proposed system for its ability in correlating various packet to provide security analyst with useful insights of the network security. Many attacks against SCADA systems are a sequence of individual attack steps. The derived features by the reasoning engine and the developed ontology can provide a base for correlating malicious events and detecting more complex attack scenarios. For instance, several packets can be correlated using their shared impact on the system, or their behavioural reflection on the system (e.g., causing same Modbus exceptions). Searching and extracting information from the background knowledge is performed through SPARQL queries (Harris et al., 2013). As a demonstration example, let us assume that a specific Modbus server (with IP address = 192.168.0.3) is under an attack originating from the Attacker (with IP address = 192.168.0.57). A list of all malicious packet flows between the two systems can be retrieved using the following SPARQL query:

Listing 3: SPARQL Query to retrieve all malicious packets between two nodes.

```
SELECT * WHERE {{?p ics:hasSrcIP '192.168.0.57'.
    ?p ics:hasDstIP '192.168.0.3'.} UNION {?p
    ics:hasSrcIP '192.168.0.3'. ?p ics:hasDstIP
    '192.168.0.57'.} ?p ics:hasPacketType ?
    PacketType. ?p ics:hasImpact ?impact. ?p ics
    :hasAttackVector ?AttackVector.}
```

Table 4: Final list of the correlated communication packets between the two systems.

Packet ID	Source IP	Server IP	Payload	Attack Vector	Impact	Status
MODBUS7	192.168.0.57	192.168.0.3	Modbus	FORCE LISTEN ONLY MODE	Denial of Service	Failed
MODBUS9	192.168.0.57	192.168.0.3	Modbus	FORCE LISTEN ONLY MODE	Denial of Service	Failed
MODBUS11	192.168.0.57	192.168.0.3	Modbus	FORCE LISTEN ONLY MODE	Denial of Service	Failed
TCP13	192.168.0.57	192.168.0.3	Non Modbus	Non Modbus Communication	Specification Violation	
MODBUS14	192.168.0.57	192.168.0.3	Modbus	RESTART COMMUNICATIONS OPTION	Denial of Service	Failed
MODBUS16	192.168.0.57	192.168.0.3	Modbus	RESTART COMMUNICATIONS OPTION	Denial of Service	Successful
MODBUS19	192.168.0.57	192.168.0.3	Modbus	RESTART COMMUNICATIONS OPTION	Denial of Service	Successful
TCP21	192.168.0.57	192.168.0.3	Non Modbus	Non Modbus Communication	Specification Violation	
MODBUS22	192.168.0.57	192.168.0.3	Modbus	CLEAR AUDIT DIAG REGISTERS	System Integrity	Successful
MODBUS25	192.168.0.57	192.168.0.3	Modbus	CLEAR AUDIT DIAG REGISTERS	System Integrity	Successful
MODBUS39	192.168.0.57	192.168.0.3	Modbus	REPORT SERVER INFORMATION	System Reconnaissance	Successful
MODBUS42	192.168.0.57	192.168.0.3	Modbus	REPORT SERVER INFORMATION	System Reconnaissance	Successful

The SPARQL query resulted in a total of 29 packet instances which contain both request and response Modbus messages. However, different from Snort IDS, the proposed system extracts the response message description about the command execution and combines this information with the request pair using a unique message correlation ID. Therefore, the final list of correlated packets for the selected two systems is presented in Table 4.

5 CONCLUSION

According to the aforementioned experiments and results, it is clear that the proposed ontology-based IDS (OSCIDS) is an effective tool for the detection of intrusions and malicious industrial communications. The use of ontology modelling can provide rich semantic logics in the represented intrusion knowledge. This enables advanced capabilities such as reasoning and deriving additional useful information from the existing knowledge, that is beyond the traditional IDS systems which utilise basic taxonomy representations. Furthermore, the correlation between packets or attacks can be made using flexible features that are not limited to the raw packet information (e.g., Source IP address) but can utilise the semantic meaning of the data (e.g., the impact on the system, the purpose of the command). We intend to apply the proposed approach on other industrial protocols such as DNP3.

REFERENCES

Barnett, B., Crapo, A., and ONeil, P. (2012). Experiences in using semantic reasoners to evaluate security of cyber physical systems. Technical report, GridSec.

Barry, B. I. and Chan, H. A. (2009). Syntax, and semantics-based signature database for hybrid intrusion detection systems. *Security and Communication Networks*, 2(6):457–475.

Carcano, A., Coletta, A., Guglielmi, M., Masera, M., Fovino, I. N., and Trombetta, A. (2011). A multidimensional critical state analysis for detecting intrusions in scada systems. *Industrial Informatics,IEEE Trans. on*, 7(2):179–186.

Chorąś, M., Flizikowski, A., Kozik, R., and Hołubowicz, W. (2010). Decision aid tool and ontology-based reasoning for critical infrastructure vulnerabilities and threats analysis. *4th CRITIS*, pages 98–110.

Drias, Z., Serhrouchni, A., and Vogel, O. (2015). Taxonomy of attacks on industrial control protocols. In *ICPE'15*, pages 1–6. IEEE.

Hadžiosmanović, D., Sommer, R., Zambon, E., and Hartel, P. H. (2014). semantic security monitoring for industrial processes. In *30th ACSAC*, pages 126–135. ACM.

Harris, S., Seaborne, A., and Prudhommeaux, E. (2013). Sparql 1.1 query language. *W3C*, 21.

Jena (2011). Jena–a semantic web framework for java. *Talis Systems*.

Kang, D.-H., Kim, B.-K., and Na, J.-C. (2014). Cyber threats and defence approaches in scada systems. In *16th ICACT*, pages 324–327. IEEE.

Langner, R. (2011). Stuxnet: Dissecting a cyberwarfare weapon. *Security & Privacy, IEEE*, 9(3):49–51.

Mallouhi, M., Al-Nashif, Y., Cox, D., Chadaga, T., and Hariri, S. (2011). A testbed for analyzing security of scada control systems (tasscs). In *IEEE ISGT*, pages 1–7. IEEE.

Modbus (2012). Modbus specification v1. 1b3. *Modbus Organization, Inc., April*, 26.

Morris, T. H., Jones, B. A., Vaughn, R. B., and Dandass, Y. S. (2013). Deterministic intrusion detection rules for modbus protocols. In *46th HICSS*, pages 1773–1781. IEEE.

Peterson, D. (2009). Quickdraw: Generating security log events for legacy scada and control system devices. In *CATCH'09*, pages 227–229. IEEE.

Roesch, M. et al. (1999). Snort ids. In *LISA*, volume 99, pages 229–238.

Sartakov, V. A. (2015). Ontological representation of networks for ids in cyber-physical systems. In *4th AIST*, pages 421–430. Springer.

Sayegh, N., Elhajj, I. H., Kayssi, A., and Chehab, A. (2014). Scada intrusion detection system based on temporal

behavior of frequent patterns. In *17th MELECON*, pages 432–438. IEEE.

Sheldon, F., Fetzer, D., Manz, D., Huang, J., Goose, S., Morris, T., Dang, J., Kirsch, J., and Wei, D. (2013). Intrinsically resilient energy control systems. CSI-IRW'13, pages 63:1–63:4. ACM.

Surridge, M., Chakravarthy, A., Hall-May, M., Chen, X., Nasser, B., and Nossal, R. (2012). Serscis: Semantic modelling of dynamic, multi-stakeholder systems.

Zhu, B. and Sastry, S. (2010). Scada-specific intrusion detection/prevention systems: a survey and taxonomy. In *1st SCS*.