Towards Learning Normality for Anomaly Detection in Industrial Control Networks

Franka Schuster, Andreas Paul, Hartmut König

Towards Learning Normality for Anomaly Detection in Industrial Control Networks

Franka Schuster, Andreas Paul, and Hartmut König

Brandenburg University of Technology Cottbus
Computer Networks Group
Cottbus, Germany
{schuster,paul,koenig}@informatik.tu-cottbus.de

Abstract. Recent trends in automation technology lead to a rising exposition of industrial control systems (ICS) to new vulnerabilities. This requires the introduction of proper security approaches in this field. Prevalent in ICS is the use of access control. Especially in critical infrastructures, however, preventive security measures should be complemented by reactive ones, such as intrusion detection. Beginning from the characteristics of automation networks we outline the implications for a suitable application of intrusion detection in this field. On this basis, an approach for creation of self-learning anomaly detection for ICS protocols is presented. In contrast to other approaches, it takes all network data into account: flow information, application data, and the packet order. We discuss the challenges that have to be solved in each step of the network data analysis to identify future aspects of research towards learning normality in industrial control networks.

1 Motivation

Currently operators of industrial control systems (ICS) aim at optimally integrating their systems into corporate infrastructures to reduce costs. For this purpose, they started to involve common information and communication technologies (ICT) in their supervisory control and data acquisition (SCADA) systems. This results in the exposition of industrial control networks to common ICT vulnerabilities and indirect connections to public networks. Simultaneously, interoperability between devices of various vendors and different automation levels is driven forward by the introduction of open standards on control and field level of ICS. For instance, Industrial Ethernet is widely used in industrial control networks, although it lacks essential security features, such as authentication and encryption. These trends affect security of industrial networks as well as critical infrastructures, such as power plants.

For complementing existing active security measures, we investigate in the use of intrusion detection in this field. In [1] we proposed a network-based intrusion detection system consisting of multiple autonomous components, called *SCADA Intrusion Detectors* (SCIDs), as illustrated in Figure 1.

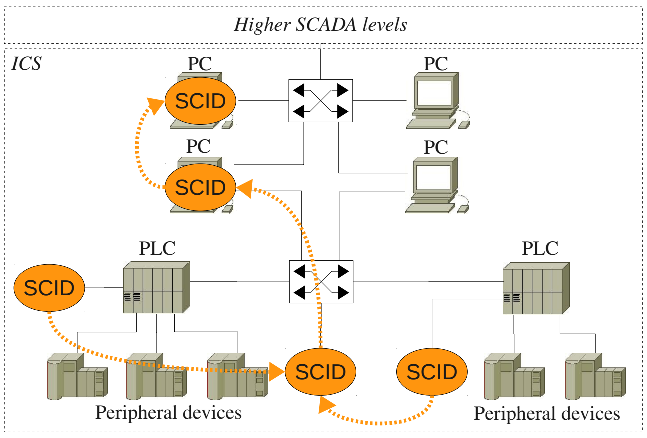

Fig. 1: Multiple SCIDs monitoring an example ICS

In contrast to common ICT networks, industrial control networks are usually characterized by a well-defined setup of devices, communication relations, and data exchanged [2] [3]. Consequently, a model of normal traffic can be defined comparatively well. Thus, each detector shall perform anomaly detection based on a model of the individual normal network traffic of its observation domain. In this realm, we investigate in the application of machine learning methods on ICS traffic to enable sophisticated anomaly detection in this field.

The main contributions of this paper are: (1) we identify the requirements for intrusion detection in ICS; (2) we present an approach for self-learning intrusion detection whose characteristics meet these requirements; (3) on the basis of this approach, we discuss the challenges and future aspects of research for realizing such a tailored self-learning intrusion detection for ICS.

The remainder of the paper is organized as follows: In Section 2 we reason the main characteristics that an intrusion detection for ICS should have. After a discussion of the drawbacks of existing works dealing with intrusion detection in this field in Section 3, we introduce an approach in Section 4 that meets all characteristics presented in Section 2. For outlining research aspects involved, we go in Section 5 into the details of each step. We conclude the paper with an outlook on our future research in this field.

2 Problem Definition

The principal problem addressed is the development of a learning approach in accordance with the intended intrusion detection. Consequently, as a prerequisite the significant criteria of the intrusion detection need to be clarified. In the following the applied detection process is reasoned by the particular characteristics of automation networks.

Network-based Analysis. Industrial control systems, especially in critical infrastructures, have to meet real-time constraints and high availability requirements. They rely on the flawless operation of their devices. Especially on field level of industrial control networks, embedded devices, such as programmable logic controllers (PLCs) and peripheral devices, are used. These systems are dedicated to perform only a specific automation task. Due to their limited resources of computing power and memory, it is often impossible to run further applications on these systems. Moreover, vendors as well as operators usually oppose any manipulation of these devices. Against this background, host-based intrusion detection is not applicable for automation devices in practice. Network-based intrusion detection, in contrast, can be integrated comparatively easily into existing automation networks. Even on control and field level this kind of intrusion detection can be placed without any manipulation of existing devices, e.g., by listening to a mirror port of a network switch.

Deep Packet Inspection. In high-speed networks flow-level monitoring is increasingly outracing packet-based analysis because the monitoring systems lack processing power and storage capacity necessary for a deep packet inspection for the corresponding data rates [4]. While flow-level analysis is intended to handle huge data rates by abstracting from detailed packet data, this limitation is neither required nor suitable for analyzing automation data due to the following reasons. First, packet-based analysis of automation traffic does not require extensive resources, because the amount of data is far below [5] what a conventional deep packet inspection can process (>100 Mbit/s). Second, flow-based monitoring generally omits payload analysis which is essential for detecting protocol-specific attacks, such as Man-in-the-Middle attacks on PROFINET IO [6] or *false data injection* in general [7] [8]. Without a deep packet inspection it is not possible to distinguish between packet types of the automation protocol used (e.g., read requests from write requests) and thus communication cannot be analyzed regarding anomal packet sequences between automation devices. For these reasons, packet-based analysis is preferred to flow-level analysis. Nevertheless, we also analyze packet data during packet-based analysis that is usually considered in flow-based analysis, i.e., the monitoring of communication relations.

N-gram Anomaly Detection. Today the specifications of most Ethernet-based ICS protocols are officially available. With protocol-specific knowledge, an attacker may launch attacks either by interferring normal protocol sequences by additional packets (*sequence-based attacks*) or by manipulating data of packets within a legitimate sequence (*content-based attacks*), or both. Whereas single-packet anomaly detection can help to detect content-based attacks, such as false data injection attacks, identification of sequence-based attacks requires to monitor sequences of packets. Examples for such sequence-based attacks are the aforementioned Man-in-the-Middle attack on a PROFINET IO setup [6] or Denial-of-Service attacks by *packet flooding* on the MODBUS TCP protocol [9] and DNP3 over TCP [10]. Since these attacks can be triggered by packets that satisfy the protocol-specific packet formats and contain ordinary data, these attacks cannot be detected by a single-packet analysis.

The feasibility to deploy n-gram analysis in real environments is addressed in [11], where the homogeneity of ICS traffic is outlined to be a key issue for a high detection capability and a low rate of false positives. In our approach each gram refers to the result of the deep packet inspection of a network packet. Consequently, an n-gram characterizes a specific sequence of n monitored packets. While learning sequences of packets as n-grams, a packet that, if considered isolated, looks ordinary can be identified as anomal in an unusual packet sequence.

Unsupervised learning. Machine learning can be subdivided into *supervised* and *unsupervised learning*. In supervised learning the input data for learning are labeled, e.g., assigned to classes. The task is to learn a model to predict the class for new data. In line with this, supervised learning for intrusion detection is done by learning normal data as well as attacks in order to apply *misuse detection*. The input data of unsupervised learning, in contrast, are unlabeled and the aim is to find a model for a representation of the input data. Associated with this approach is the idea of anomaly detection: a model representing normality is learned from unlabeled normal data to be able to identify attacks as a kind of anomalies. Both approaches have been subject for research in network-based intrusion detection. If unknown attacks can be expected, however, it has been empirically shown in [12] that unsupervised methods are more qualified for practical purposes, because their detection capability is similar to supervised learning, while they do not require the tedious preparation of labelling the input data. Consequently, unsupervised learning is the most promising method also for learning normal traffic for anomaly detection in ICS.

3 Related Work

After reasoning why network-based anomaly detection using a learned model of normal traffic is the most suitable kind of intrusion detection for ICS, we will focus the discussion of related work on similar approaches.

In [13] [14] contributions to a state-based IDS are presented. The system performs anomaly detection based on a decision whether the monitored system enters a critical state. For this purpose, a central virtual image of the physical state of the whole system is set up and regularly updated.

The presented algorithm in [15] uses deep packet inspection and estimates if a network packet has anomal effect on a memory variable of an ICS device. This approach, however, requires both detailed understanding of the used ICS network protocol and extensive knowledge about variables stored in the RAM variable memory of all monitored PLCs of the ICS.

Other approaches apply *Artificial Neural Networks* to perform anomaly detection in ICS. The authors of [16] also focus on n-gram anomaly detection, where each gram refers to the attribute extraction from a network packet. In [8] a back-propagation algorithm is used to build the neural network for a network-based intrusion detection system. Although these works provide relevant contributions, both are based on supervised learning that depends on labeled input data, i.e., requires normal as well as attack data.

The anomaly detection process in [17] relies on pattern matching. It combines *Autoassociative Kernel Regression* for model generation with a binary hypothesis technique called *Sequential Probability Ratio Test* while detection. The proposed kind of model generation, however, relies on the assumption that security violations are reflected by a change in system usage, which is subject of the detection. This obviously limits the detection capability.

Research in flow-based anomaly detection for ICS is motivated in [3]. The model generation focuses on finding relations between network flows based on clustering and correlation. Here, we argue the limitation of a detection that only focuses on flow data analysis, as we have explained in the previous section.

4 Learning Approach

The model generation and anomaly detection of our approach focuses on the following data:

– **Communication relations:** Communication relations refer to network flows. These are sequences of packets from a source to a destination device using a certain protocol. For monitoring communication relations between ICS devices, flow data, i.e., source and destination addresses as well as the used protocol, have to be determined. This also allows to gather flow characteristics, such as byte or packet number transmitted in a certain time interval.
– **Integrity of ICS application data:** Beyond the monitoring of communication relations, the actual data exchanged are subject of the monitoring. For this purpose, the payload of network packets is inspected and protocol-specific data are analyzed regarding anomalies.
– **Consistency of the packet exchange:** Based on identified flows and knowledge about the used protocol, the type of each packet within a flow can be determined. Thus, the order of packets exchanged in a communication relation can be evaluated.

The principle for learning this information from the industrial control network is depicted in Figure 2. Initially, network packets are captured and decoded by a deep packet inspection (DPI) sensor that is capable to analyze packets of the used ICS protocols. From each packet a set of attributes, so-called *features*, is extracted. For the later application of mathematical operations in the machine learning stage, this set of features, i.e., list of original packet attributes, are mapped to a vector of real numbers (*feature vector*). In this process of feature conversion a suitable numerical representation of the feature values has to be found with respect to the feature types (e.g., categorical or continuous) and dependencies between the features. In the next step the current feature vector is aggregated with the feature vectors of the $n - 1$ previously monitored network packets. The resulting n-gram represents an input instance for the machine learning algorithm applied. Finally all information mentioned above is concentrated within the n-grams as input for the machine learning procedure.

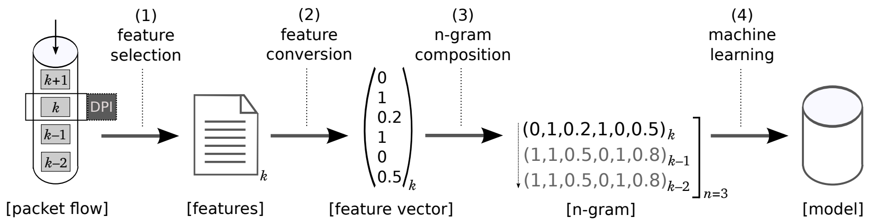

Fig. 2: Schematic depiction of the learning approach

The approach can be applied in two ways: (1) either learning is realized protocol-related, so that for each ICS protocol used in the network a separate model of normality is learned based on a protocol-specific set of features, or (2) a common feature set for all supported ICS protocols is defined and used to learn a shared model. The comparison of both strategies is an interesting issue for future investigation, which represents aspect 1 for future research. In the following we will denote further aspects using consecutive numbering. The discussion in this paper, however, focuses on the first strategy, since it promises a more tailored learning and anomaly detection for the respective protocols.

5 Challenges of Learning for ICS Intrusion Detection

Development and implementation of a self-learning anomaly detection for ICS induces a set of challenges. We address these challenges based on the steps of the approach introduced in Section 4. If necessary we exemplify our explanations using the example of the ICS protocol Profinet IO.

5.1 Understanding Automation Protocols

The development of a deep packet inspection for application on control and field level of industrial control networks requires to understand the protocols spoken on these levels. The aim is to extract the data of each network packet and to map them to a representation for further analysis. For such a packet-based analysis, a protocol-specific decoding of network packets has to be implemented. Here, it also has to be regarded that different ICS protocols expect different protocol stacks for transportation. For instance, whereas Profinet IO bypasses the network and transport layer, Modbus TCP requires the regular TCP/IP stack. Thus, the specification of the respective protocol has to be analyzed in advance regarding transportation stack and message formats. This work has to be done individually for each ICS protocol that the intrusion detection shall be capable to support. Nevertheless, the effort for realizing such a protocol analysis is well spent, because it also allows a vulnerability assessment and the derivation of possible attacks. This is also fruitful for a later evaluation of the implemented anomaly detection. Since traffic capturing and decoding is an essential feature of packet-based intrusion detection systems in general, existing solutions [18] can be extended by the respective protocol knowledge to appropriate sensors.

5.2 Feature Selection

The challenging part in this step is to decide which ICS protocol data are worth and suitable to be learned. Since our approach shall not be limited to learning the traffic of a certain protocol, but rather be applicable for a wide range of Ethernet-based ICS protocols, we abstract from the various protocol data formats and focus our explanation here on data that are usually part of Ethernet-based ICS protocols:

– **source and destination addresses** – unique identifiers of sender and receiver (e.g., MAC addresses),
– **protocol type** – the identifier of the ICS protocol; in case of PROFINET value 0x8892 encoded in the Ethernet frame's *EtherType* field,
– **packet type** – the type of protocol-specific packet that the Ethernet frame conveys; in case of PROFINET this can be for example a *DCP request* or *DCP response* packet for device identification or an alarm packet [19],
– **packet data** – the ICS application data, e.g., parameters for cyclic control.

From each monitored frame at the respective network interface, the deep packet inspection sensor constructs a protocol-specific object containing the mentioned information in form of a set of features. While this object in detail depends on the protocol-specific fields, here a generic high-level description of this object is chosen, which in the following is referred to as *feature object.*

```
struct featureObj {
  timestamp tstamp;         (1)
  identifier source;        (2)
  identifier destination;   (3)
  enum event_type type;     (4)
  union event_data event;   (5)
}
```

The feature `tstamp` holds a value used to recover the temporal order of packets, whereas `source` and `destination` encode addresses of the sending device and the one receiving the packet. The categorical feature `type` defines one of the standard packet types of the ICS protocol. The remaining feature `event` contains all packet-type-specific data, e.g., parameters for control or feedback data transmission between ICS devices.

Features (2-3) help to identify (unidirectional) flows and (bidirectional) communication relations between automation devices. Based on features (1-4), the concrete sequence of packet types exchanged between these devices can be monitored. This is sufficient for detecting, for instance, the Man-in-the-Middle attack presented in [6] as anomaly from learned normal sequences. By the use of detailled features contained in the complex feature (5), an anomaly detection based on learned normal operation data can be realized. If, for instance, a parameter like a valve pressure measurement contained in (5) normally varies within the boundaries of interval $[a, b]; a, b \in \mathbb{R}$ then a value $x \in \mathbb{R}$ with $(x << a)$ or $(x >> b)$ can be detected as anomaly from the learned normal interval.

5.3 Feature Conversion for Learning

Machine learning deals with finding characteristics in provided *training data* and generalizing from these characteristics for evaluation of new, unseen data instances in *test data*. For this purpose, machine learning relies on the use of mathematical constructs and algorithms. Hence, data instances for learning have to be converted into numerical representations and summarized in a feature vector. Finding an optimal representation for the input data as numerical data is actually a key issue for successful application of machine learning in general.

The conversion applied depends on the type of feature. Each feature that has been extracted from a packet is either of *categorical*, *identifying*, or *continuous* type. Table 1 summarizes the attribution and provides some example values for features contained in the introduced feature object.

Table 1: Feature types and example values

Feature	Type	Example values
`tstamp`	continuous	1253306964; 1361483071
`source`	identifying	ac:de:48:00:00:80; 00:80:41:ae:fd:7e
`destination`	identifying	d6:6c:51:84:af:bc; 84:2b:2b:92:41:a8
`type`	categorical	*read request*; *write request*; *alarm*
`event`	(complex)	(`param1`=3659, `param2`=0.85, `param3`=7)

The process of converting a set of features into a numerical representation basically consists of two steps: (1) *mapping* each feature to a value in real space, i.e., a vector in $\mathbb{R}^n; n \in \mathbb{N}$, and (2) *scaling* the real values of features to lie in similar range. Scaling has two advantages for learning: First, if all features are represented by real values in similar range, no features in greater numerical ranges can dominate those in smaller numerical ranges while the application of mathematical operations during learning. Second, numerical problems during calculation are avoided. For instance, some machine learning methods [20] apply kernels that depend on the inner products of feature vectors which might be difficult to determine in case of highly varying feature spaces.

Mapping Categorical Features. The main criterion of a categorical feature is the dimension $n \in \mathbb{N}$ of possible categories. The conversion can either be realized by just defining a real number for each category (e.g., integers 1 to n) or by constructing an n-dimensional vector whose values at position $i \in \{1 \dots n\}$ are defined as

$$v(i) = \begin{cases} 1, & \text{feature value is of the } k\text{-th category} \\ 0, & \text{otherwise} \end{cases} ; k \in \{1 \dots n\} \; . \qquad (1)$$

In terms of the introduced feature object, if feature `type` can be one of the values $\{$*read request*, *write request*, *alarm*$\}$, then conversion of feature value *read request* would result in $(1, 0, 0)$, feature value *alarm* correspondingly in $(0, 0, 1)$. Comparing both ways of converting a categorical ICS feature in the context of the machine learning method applied is another relevant issue of study (aspect 2).

Mapping Identifying Features. These are features holding addresses or other identifiers, such as device names, that are a typical aid for ICS operators to distinguish and locate automation devices. Learning concrete values of an identifying feature, e.g., the integer value of a MAC address' byte sequence, is not useful. It would result in a model of normality, in which new MAC addresses with similar integer values like normal addresses would be also considered as normal. Even they, however, explicitely identify a new device, which is, in terms of homogenous ICS traffic, an anomal event.

Instead, identifying features have to be converted in a way that the applied learning method results in a model that only characterizes the identifiers as normal that have explicitely seen during learning phase. Thus, a better way of converting an identifying feature is to allow a fixed maximum number $n \in \mathbb{N}$ of devices in the monitoring domain and to store each seen identifier in the training data in a list of length n. Then, conversion of a specific identifier is realized like conversion of a categorical feature, where the list position of the identifier is handled like a category (see Formula 1). For illustration: If in a monitored network packet in training or test data the identifying feature `source` contains MAC address y while MAC addresses (x, y, z) have already been seen, then y would be converted to $(0, 1, 0)$.

Mapping Continuous Features. In the realm of ICS, most operation parameters are provided as real numbers. This, for instance, can be a measured value as part of a feedback packet from a peripheral device to a PLC. Such a parameter would be part of complex feature `event`. In contrast to categorical and identifying features, the concrete value of this feature has to be learned in order to identify anomalies in ICS application data.

Some machine learning methods rely on discrete input data. Consequently, continuous features have to be discretized for these algorithms. In [21] a comprehensive overview about existing approaches and an empirical comparison of discretization for continuous attributes is provided. In terms of this work, *unsupervised discretization* methods are suitable for our approach. More recent discussions in this field can be found in [22] and [23].

If continuous features in the form of real numbers are accepted as input for the applied machine learning method (e.g., methods in [20]) the mapping step during conversion can obviously be omitted. Here, scaling is crucial for successful learning. Nevertheless, discretization may also be supportive of learning even if the learning algorithm would accept continuous data. This, however, can only be evaluated as a kind of preprocessing in strong connection with the concrete learning algorithm applied (aspect 3).

Scaling. In the presented ways for converting categorical and identifying attributes the kind of mapping applied implicitely involves also scaling, since the length of the resulting vector is always maximum 1 (It can also be 0 in case the value is not defined for the categorical respectively identifying feature, which results in conversion to a *null vector*).

If continuous features are not discretized for learning, they have at least to be scaled. This involves the transformation of real values into a smaller interval. It can be realized by defining the minimum and maximum value accepted for this feature and linear scaling to a smaller interval, such as $[0, 1]$ or $[-1, +1]$. The same scaling method, however, should be applied for the respective feature during the training and anomaly detection phase. For example, if the feature `param1` in abstract feature `event` has been scaled from $[0, 10000]$ to $[0, 1]$ during training, then during anomaly detection a value 3659 for `param1` has to be scaled to 0.3659. Finding the right scaling approach, i.e., scaling range for each feature and ratio across features, is a further aspect of study to optimize self-learning anomaly detection for ICS (aspect 4).

Feature Dependencies. It might be the case that there are dependencies between features for some ICS protocols which have to be explicitly regarded during conversion. One approach for handling dependencies is to construct a complex feature value from values of depending features and to learn this aggregated feature value instead of the individual ones. For example, if the range of feature `param3` in complex feature `event` by protocol specification depends on the value of feature `type`, then this could be expressed as follows during conversion: A simple method is to map `param3` to a real number, so that its concrete value only affects the less significant digits, whereas the value of `type` dictates the more significant ones. Thus, instead of just scaling, real value 7 would be mapped beforehand to real number $x007$ where x represents a digit based on the k-th category for categorical value `type`. As illustrated by this simple example, exploring sophisticated dependency-based conversion methods is a further subject of investigation (aspect 5).

5.4 Evaluation of Learning Algorithms

Besides the optimal choice of parameter n (aspect 6) also the choice of the unsupervised learning algorithm for generating the normal traffic model, based on the n-grams of converted features, is the most relevant aspect towards the implementation of a sophisticated self-learning anomaly detection for ICS (aspect 7). We plan to evaluate the algorithms regarding their behaviour on ICS data:

- **efficiency** of the algorithm, i.e, the number of learning examples necessary for a certain detection accuracy on new, unseen data,
- **stability** of the learned model to variations of input parameters,
- **scaling** of the learning effort with the number of training instances and input features.

In this context, it will be interesting to find out whether a specific learning algorithm can distinctly outperform the other ones or if the learning success will be similar among different algorithms. Another aspect of study will be the prevention of overfitting in the learning (aspect 8).

6 Final Remarks

In this work we have presented an approach for learning normal ICS traffic to support anomaly-based intrusion detection in this field. In contrast to existing methods, our approach combines the learning of communication relations, ICS operation data as well as exchanged packet sequences. By explaining the steps of the approach, we identified eight aspects that affect the quality of learning the addressed information. In general, the application of machine learning techniques for ICS security is a very promising field. For successfully applying machine learning and anomaly detection in ICS networks, however, the identified aspects for optimizing the learning have to be explicitly evaluated with regard to ICS traffic characteristics. We address ourselves to this task. So we plan to apply several machine learning algorithms as part of our SCADA intrusion detector in order to investigate in the identified aspects for proper learning. Analysis will first focus on monitoring a Profinet IO network. For this purpose, we have implemented a Profinet-specific deep packet inspection sensor. In [24] we have identified numerous vulnerabilities and possible attacks on the protocol which will help us to prove the detection accuracy of our approach.

References

1. Schuster, F., Paul, A.: A Distributed Intrusion Detection System for Industrial Automation Networks. In: Proc. of the 17th IEEE Intl. Conf. on Emerging Technologies and Factory Automation (ETFA 2012). IEEE (2012)
2. Hadziosmanović, D., Bolzoni, D., Etalle, S., Hartel, P.H.: Challenges and Opportunities in Securing Industrial Control Systems. In: Proc. of the IEEE Workshop on Complexity in Engineering (COMPENG 2012). IEEE (2012)
3. Barbosa, R.R.R., Pras, A.: Intrusion Detection in SCADA Networks. In: Stiller, B., Turck, F. (eds.) Mechanisms for Autonomous Management of Networks and Services (AIMS 2010), LNCS, vol. 6155, pp. 163–166. Springer (2010)
4. Hofstede, R., Pras, A.: Real-Time and Resilient Intrusion Detection: A Flow-Based Approach. In: Dependable Networks and Services (AIMS 2012). LNCS, vol. 7279, pp. 109–112. Springer (June 2012)
5. Barbosa, R.R., Sadre, R., Pras, A.: Difficulties in Modeling SCADA Traffic: A Comparative Analysis. In: Taft, N., Ricciato, F. (eds.) Proc. of the 13th Intl. Conf. on Passive and Active Measurement, LNCS, vol. 7192, pp. 126–135. Springer (2012)
6. Åkerberg, J., Björkman, M.: Exploring Security in PROFINET IO. In: Proc. of the 33rd Annual IEEE Intl. Computer Software and Applications Conf. (COMPSAC 2009). IEEE (2009)
7. Liu, Y., Ning, P., Reiter, M.K.: False Data Injection Attacks Against State Estimation in Electric Power Grids. In: Proc. of the 16th ACM Conf. on Computer and Communications Security (CCS 2009). ACM (2009)

8. Gao, W., Morris, T., Reaves, B., Richey, D.: On SCADA Control System Command and Response Injection and Intrusion Detection. In: Proc. of the Fifth eCrime Researchers Summit. pp. 1–9. IEEE (2010)
9. Nai Fovino, I., Carcano, A., Masera, M., Trombetta, A.: An Experimental Investigation of Malware Attacks on SCADA Systems. Intl. Journal of Critical Infrastructure Protection 2(4), 139–145 (2009)
10. Jin, D., Nicol, D., Yan, G.: An Event Buffer Flooding Attack in DNP3 Controlled SCADA Systems. In: Proc. of the 2011 Winter Simulation Conf. IEEE (2011)
11. Hadziosmanović, D., Simionato, L., Bolzoni, D., Zambon, E., Etalle, S.: N-gram Against the Machine: On the Feasibility of the N-gram Network Analysis for Binary Protocols. In: Proc. of the 15th Intl. Symposium on Research in Attacks, Intrusions and Defenses (RAID 2012). LNCS, vol. 7462, pp. 354–373. Springer (2012)
12. Laskov, P., Düssel, P., Schäfer, C., Rieck, K.: Learning Intrusion Detection: Supervised or Unsupervised? In: Roli, F., Vitulano, S. (eds.) Image Analysis and Processing, LNCS, vol. 3617. Springer (2005)
13. Carcano, A., Fovino, I.N., Masera, M., Trombetta, A.: State-Based Network Intrusion Detection Systems for SCADA Protocols: A Proof of Concept. In: Proc. of the Fourth Intl. Workshop on Critical Information Infrastructures Security (CRITIS 2009). LNCS, vol. 6027, pp. 138–150. Springer (2009)
14. Carcano, A., Coletta, A., Guglielmi, M., Masera, M., Fovino, I.N., Trombetta, A.: A Multidimensional Critical State Analysis for Detecting Intrusions in SCADA Systems. IEEE Trans. on Industrial Informatics 7(2), 179–186 (2011)
15. Rrushi, J., Kang, K.D.: Detecting Anomalies in Process Control Networks. In: Palmer, C., Shenoi, S. (eds.) Critical Infrastructure Protection III, IFIP Advances in Information and Communication Technology, vol. 311. Springer (2009)
16. Linda, O., Vollmer, T., Manic, M.: Neural Network based Intrusion Detection System for Critical Infrastructures. In: Proc. of the 2009 Intl. Joint Conf. on Neural Networks (IJCNN 2009). pp. 1827–1834. IEEE (2009)
17. Yang, D., Usynin, A., Hines, J.W.: Anomaly-based Intrusion Detection for SCADA Systems. In: Proc of the Fifth Intl. Topical Meeting on Nuclear Plant Instrumentation, Control and Human Machine Interface Technologies (NPIC/HMIT 2006). pp. 12–16. Curran Associates (2006)
18. Snort: Snort 2.9.4 (www.snort.org)
19. Neumann, P., Pöschmann, A.: Ethernet-based Real-time Communications with PROFINET IO. In: Proc. of the Seventh WSEAS Intl. Conf. on Automatic Control, Modeling and Simulation (ACMOS 2005). pp. 54–61. World Scientific and Engineering Academy and Society (WSEAS) (2005)
20. Schölkopf, B., Smola, A.J.: Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond. MIT Press, Cambridge (2001)
21. Dougherty, J., Kohavi, R., Sahami, M.: Supervised and Unsupervised Discretization of Continuous Features. In: Proc. of the Twelfth Intl. Conf. on Machine Learning (ICML 1995). Morgan Kaufmann (1995)
22. Liu, H., Hussain, F., Tan, C.L., Dash, M.: Discretization: An Enabling Technique. Data Mining and Knowledge Discovery 6(4), 393–423 (2002)
23. Peng, L., Qing, W., Yujia, G.: Study on Comparison of Discretization Methods. In: Proc. of the Intl. Conf. on Artificial Intelligence and Computational Intelligence (AICI 2009). IEEE (2009)
24. Paul, A., Schuster, F., König, H.: Towards the Protection of Industrial Control Systems – Conclusions of a Vulnerability Analysis of Profinet IO. Accepted for the 10th Conf. on Detection of Intrusions and Malware and Vulnerability Assessment (DIMVA 2013)