Providing SCADA network data sets for intrusion detection research

Antoine Lemay and José M. Fernandez, *École Polytechnique de Montréal*

Abstract

High profile attacks such as Stuxnet and the cyber attack on the Ukrainian power grid have increased research in Industrial Control System (ICS) and Supervisory Control and Data Acquisition (SCADA) network security. However, due to the sensitive nature of these networks, there is little publicly available data for researchers to evaluate the effectiveness of the proposed solution. The lack of representative data sets makes evaluation and independent validation of emerging security solutions difficult and slows down progress towards effective and reusable solutions.

This paper presents our work to generate representative labeled data sets for SCADA networks that security researcher can use freely. The data sets include packet captures including both malicious and non-malicious Modbus traffic and accompanying CSV files that contain labels to provide the ground truth for supervised machine learning.

To provide representative data at the network level, the data sets were generated in a SCADA sandbox, where electrical network simulators were used to introduce realism in the physical component. Also, real attack tools, some of them custom built for Modbus networks, were used to generate the malicious traffic. Even though they do not fully replicate a production network, these data sets represent a good baseline to validate detection tools for SCADA systems.

1. Introduction

The scientific method is based on the ability to empirically test predictions. This is achieved by gathering data on observable phenomenon and using this data to test a hypothesis. As such, the quality of available data is an important factor in the quality of research. In fact, in his critique of IDS evaluation methodology, McHugh [1] identifies the DARPA data sets prevalently used for testing IDS as a significant liability for the validity of the results.

While the lack of data sets is a problem in a number of security fields, the field of industrial control systems (ICS) network security is particularly affected. Many of these ICS networks are used to control elements of critical infrastructure. As such, operators of ICS networks are reluctant to provide data from production systems. The fact that ICS networks are notorious for poor security further enhances the perception of risk in making data from operational ICS networks publicly available.

Because of the lack of publicly available data, researchers wanting to improve security of ICS networks have to resort to generating their own data or use data sets shared under confidentiality agreements. Under these circumstances, it is not possible to provide an external validation of the results obtained in the research, even if the data was highly representative of real ICS networks.

We believe that the availability of public data sets for ICS network security research is an important step toward improving the quality of results in the field. This paper presents data sets for supervisory control and data acquisition (SCADA) network security research, which is a subset of ICS network security research. The data sets include captures for a number of different configuration parameters and a number of different attacks requiring varying degrees of sophistication for detection.

Furthermore, the data sets include both the full packet captures and a file containing the labeling of malicious traffic. This provides researchers complete knowledge of the ground truth while retaining the ability to extract any feature they can imagine.

The paper also provides detailed information on how the data sets were generated. It also identifies the principal biases that can be introduced by limitations in the traffic generation approach used.

The paper begins with background information on SCADA systems. Then, it presents an overview of the types of data sets used in literature and their limitations. Next, the paper describes the major challenges in generating data sets for SCADA networks. The paper then provides details on the data set generation process and describes contents of the data sets we generated. Finally, the paper offers a brief conclusion.

2. Background

In order to understand the data sets provided, it is necessary to possess some background knowledge on field controllers and SCADA systems, in particular, and on the general architecture and normal operation of Modbus networks.

In many enterprises, the ICS networks are organized in a more-or-less standard topology that is described by the Purdue Enterprise Reference Architecture model. This model lays out the components in different levels.

The physical process is at Level 0. The intelligent field devices, i.e. sensors, actuators and their controllers, are situated at Level 1. Finally, higher-level controls, such as SCADA systems, are located at Level 2. Using these guidelines, all Level 1 equipment is grouped together on the same plant LAN. Similarly, all Level 2 equipment is grouped in the control center LAN. This produces an architecture similar to the one presented in Figure 1.

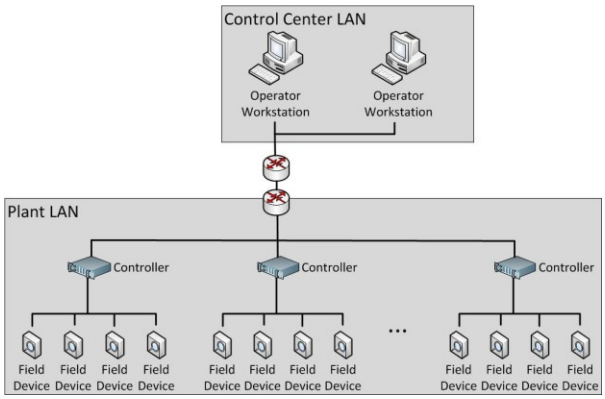

Figure 1. Architecture of a Modbus system

In the setup used in this paper, each operator workstation hosts SCADA software. This software consists of two main components: the Master Terminal Unit (MTU) component and the Human Machine Interface (HMI) component. The MTU is responsible for maintaining information about the state of the physical process. In a Modbus implementation, this is done by continuous polling of each controller. Through this polling, an update is requested on the state of all field devices attached to the controller. Every few seconds (the exact polling interval value is configurable) the MTU component sends a Modbus read packet to each controller. The controller then responds with a read response packet containing all requested values. This will be dubbed *polling traffic* in the rest of the paper. In the case of distributed systems, a Remote Terminal Unit (RTU) may aggregate local values and handle communication with the MTU.

The Human Machine Interface (HMI) component provides the operator with a visualization of the state of the physical process based on the most up-to-date measurements collected by the MTU component. It also allows operators to alter the state of the system through a graphical interface. Whenever an operator performs an action, the MTU component sends a Modbus write request to the controller supervising the targeted field device. The controller then alters the state of the field device and sends back a write response packet.

This architecture implies that the majority of the traffic present on SCADA networks is automated traffic, driven by communications between the SCADA MTU and the field controllers. Humans interact with the system only when they wish to alter the state of the industrial process being controlled.

3. Use of SCADA Data sets in the literature

The properties of SCADA systems make them more deterministic than common IT networks. This determinism can be leveraged in a number of ways to improve defenses. Notably, the increased regularity observed in these systems can be leveraged to improve detection of intrusions. This is possible by building on the knowledge of static configuration files [2] or by studying the traffic statistical properties [3].

However, research in the field of intrusion detection for SCADA systems suffers from the lack of viable data sets. For example, Hadžiosmanović *et al.* [4] have found SCADA protocols to be well adjusted to anomaly detection based on *n*-grams. But, they comment there is less publicly available attack data for SCADA protocols than for traditional IT protocols. In fact, the particular data set used for attack traffic in their *n*-grams work employed data used by DigitalBond to develop the Quickdraw rule-based IDS which is heavily slanted toward malformed packets. We believe this could bias their results toward a single attack class.

Another option to obtain data sets for experimentation is to use traffic from a real SCADA deployment. While SCADA network operators seldom provide access to production data, some researchers have managed to gain access. Notably, the work of Barbosa *et al*. [5, 6] uses traffic from a water plant deployment. However, when using live data sets, it can be very difficult to know the ground truth as unwanted traffic may be present in the data set. Manual inspection of large data sets is often required to validate the contents. Even if efforts are invested in validating the data set, often the data cannot be shared by the researchers, making it impossible for other researchers to build on the initial results or to independently validate the results.

For SCADA network research, the apparent similarity with traditional IT networks can yield unreasonable assumptions. We can cite the the work of Valdes and Cheung [7] in anomaly detection as an example. In this work, they attempted to use packet length and interarrival time for anomaly detection. Unfortunately, they calculate and present their results based on T-test scores. This assumes that the features follow a normal distribution, which is not the case. In fact, because traffic is generated by machines performing the same actions

repeatedly, the statistical properties of traffic are heavily weighted toward a small subset of values [3]. While their paper was sufficiently explicit to validate their conclusions, many other papers lack the details required to validate their claims.

As such, we feel it is useful to provide a baseline data set that can be publicly used, analyzed and discussed for the purpose of doing network intrusion detection research.

4. Challenges

To generate data sets suitable for network intrusion detection research, a number of challenges must be addressed. This section describes how we addressed injection timing problems, information properties issues and labeling conflicts.

4.1. Injection timing

One of the most common methods to generate data sets for intrusion detection is to take a known-good traffic capture and inject malicious traffic in it. This technique has the advantage of being able to generate base traffic containing a lot of noise and to be able to know the ground truth to validate results. However, because the injected attack traffic and the background traffic come from different packet captures, it may be easier to identify the attack traffic if appropriate care was not taken when merging the captures.

This problem is particularly important when generating traces for SCADA traffic. As discussed in Section 2, most SCADA protocols are polling based, with the MTU sending a polling request at pre-defined intervals. This means that the timing between packets is fairly rigid. As such, great care must be taken when integrating packets from other captures to make sure that the time scaling is accurate.

To illustrate this point, consider that the traffic captured from the Modbus storage-based covert channel operates on a different time scale than the polling configured for regular SCADA traffic. The covert channel used to generate the data sets sends packets outside of normal polling intervals in order to increase the data rate at the expense of stealth. As such, it would be relatively easy to differentiate between covert channel traffic and regular traffic using timing metrics. However, any change in configuration for either the attack tool or the SCADA network could drastically change the validity of the results.

In our work, we have chosen to sidestep this issue by using actual implementations of the attacks on the SCADA network. This ensures maximal fidelity of the network timing. Packet captures containing no attack traffic are also provided. The only exception is the covert channel traffic which was generated on a separate network that contained no background noise traffic to avoid TCP socket conflicts.

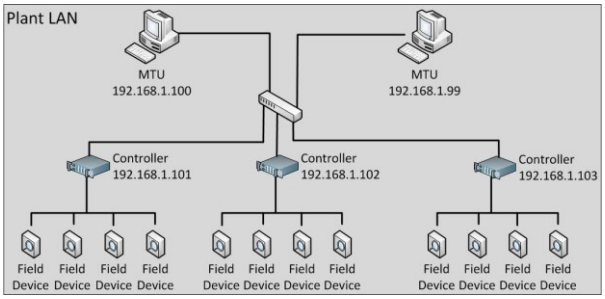

Figure 2. Example of a SCADA network used in data set generation (2 MTU and 3 Controllers)

4.2. Information properties of the traffic

Another challenge arises when providing data sets that are used with deep-packet inspection approaches. In particular, for SCADA systems, a number of approaches use physical properties of the industrial process being controlled to perform anomaly detection. For an example of this kind of approach, see Gao *et al*. [8].

In reality, the content of the packets are affected by a number of properties of the system. For example, the number of sensors may affect the number of values that are transferred from a controller to the MTU. Similarly, the amount of noise in the physical system may affect the entropy of the distribution of measurement values provided by the sensors. For example, a safety valve, which is turned on only in times of emergency, has different information properties than those of a sensor measuring the voltage on a line during a period of high solar flare activity. In these cases, the exact modelling of the physical system may have a direct impact of the validity of results that take into account the entire contents of the packet of the state of the physical system.

In our work, we have partially addressed this issue by using a simulator to drive the response from the controllers, thus adding a realistic level of entropy into the system. The system simulated on each controller is illustrated at Figure 2.

In this simulation, each controller represents a small electrical network powered by a 12 000 Volt source. The network consists of one main supply branch and three sub-branches, A, B and C. Each branch, including the main branch, incorporates a cut-off breaker. The controllers provide measurements on the voltage of each branch to the MTU. These voltages are obtained

by calculations performed by the simulation code. This enables the values included in Modbus traffic to change depending on the status of the breakers, introducing some realism in the Modbus message contents.

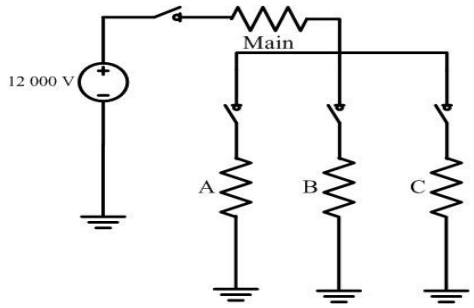

Figure 3. Line diagram of simulated system

4.3. Labeling

Another challenge is the labeling of malicious and non-malicious traffic. In our case, because we generate all the traffic ourselves, we have perfect knowledge of the ground truth, circumventing issues arising from dealing with data sets from production networks. However, this does not mean that there are no difficulties related to labeling.

The main concern comes from how "malicious" is defined. Let us consider an attacker that connects to a FTP service and sends an exploit. This is clearly malicious behavior and should be labeled as such. However, when we dig deeper in the packet capture, questions can arise. For example, it would be possible to argue that the TCP handshake required to connect to the service to send the exploit is not malicious. After all, it is no different than the other handshakes performed by legitimate clients. In that case, only the packet containing the actual exploit should be labeled malicious. However, in that case, how should attacks that abuse functionality be labeled? As an example, under a labeling discipline where only the packets containing attack code are labeled malicious, none of the packets related to a port scan should be labeled malicious. After all, a port scan only performs legitimate (or half-complete) handshakes.

This ambiguity is particularly important with some of the types of attacks that were implemented in our data set. These attacks abuse the lack of authentication on SCADA protocols to inject commands or to fingerprint the content of a controller using properly formed packets.

To address this ambiguity, the following labeling discipline was used: if a packet is part of a conversation that includes malicious activity, it is labeled malicious. Otherwise, it is labeled non-malicious. For example, a Modbus polling packet from the MTU is labeled non-malicious, but a packet containing an exploit is labeled malicious. However, with our labeling discipline, the SYN packet that establishes the connection containing the exploit packet is also labeled malicious, even if it does not contain any active payload per se.

5. Data set generation

This section will discuss how the data sets were generated and what they contain. It starts by providing information on the SCADA network data set, and then information related to SCADA covert channel command and control data.

All the data sets discussed in this section are available online at the following URL:

https://github.com/antoine-lemay/Modbus_dataset.

5.1. SCADA network data set

For the SCADA network data set, a small SCADA network was implemented in a SCADA sandbox [9]. Modbus was chosen as the exemplar for SCADA networks because of the better availability of open source tools supporting the protocol. Moreover, we chose the Modbus variant that closes TCP connections after each request instead of the variant with a long keep-alive time. This network was comprised of a varying number of MTUs, implemented using ScadaBR [10], and a varying number of controllers, implemented using Modbus_tk [11]. The exact number of MTU and RTU is varied in order to provide an illustration of the impact of different implementations on the traffic. An example network which includes 2 MTU and 3 controllers is included in Figure 3. All of the software runs on Windows XP machines, which, according to our discussions with SCADA network operators, is still largely used in a number of operational SCADA networks.

Because all the data sets were extracted from a similar network, the packet captures from the different data sets present similar features. This enables the training of intrusion detection tools on data sets containing no attacks and testing on the data sets containing attacks or benign variations in behavior, e.g. a change in polling interval. This also reduces the amount of effort required to isolate attack traffic and inject it into data sets that do not contain malicious activity. However, we advise that anyone doing so take care to mitigate the timing problems described in Section 4.1.

Each controller operates the same physical system, described in Section 4.2. As such, each system has the same number of points. There are four points reporting the status of the breakers, implemented as Binary Inputs, four points representing the voltage on each line,

implemented as Holding Registers and four controllable points to operate the breakers, implemented as Input Coils. These points represent the field devices attached to the controller. A limitation of our setup is that the traffic between the controller and the field devices cannot be observed.

The level of similarity between controllers is representative of some real ICS networks in industry, where designs are copy-pasted to speed up installation. However, many ICS networks would include a greater diversity of field units and sensed values. This represents a limitation of the data sets generated and should be taken into account when using the data set for experimentation.

A similar level of re-use was employed for the configuration of the MTU systems. In particular, a uniform polling configuration was used across MTU and across controllers. As with the number of nodes, the polling periodicity has been varied between data sets to provide an illustration of the impact of changing the polling interval on traffic. However, the polling interval remains constant for all controllers and all MTU in a given packet capture. This uniform polling configuration is more commonly observed than the high-similarity controller configuration described previously. This is due to the polling interval being dependent on data freshness requirements of the MTU. We do recognize that some SCADA operators may have mixed polling interval configurations. This represents a second limitation of the generated data sets that should be taken into account.

The last parameter is the number of operations performed by operators through the MTU. In some of the captures, we simulate the presence of human operators in the system by sending control packets at random intervals. These control packets consist of WRITE_SINGLE_COIL (Modbus function code 5) packets instructing a randomly selected controller to set the value of a randomly selected breaker to a random value (ON or OFF). This is done both to vary the type of traffic in the system and to introduce entropy in the measurements, as altering the state of the breakers will alter the results of the measurements. Obviously, this method of generating traffic does not fully capture human behavior. This limitation also needs to be taken into account when analyzing results obtained from the data set.

Finally, a number of attack scenarios have been implemented to illustrate an attacker attempting to compromise the system. For most of the scenarios, the attack traffic comes from one of the legitimate machines. This illustrates an attacker compromising a machine and using that machine as a foothold to attack the network. This behavior is consistent with the behavior of attackers observed in the wild, notably the attackers responsible for the cyber attack against the Ukrainian power grid in December 2015 [12]. In that attack, a machine was compromised and used to send properly formed commands to controllers to create an impact in the real world (blackout). As SCADA protocols are unauthenticated, it was not necessary to use SCADA specific exploits and our data set is constructed to approximate this activity. To simulate this behavior, a Metasploit pivot point was established on a machine from an unobservable network (an additional network card was added to the compromised machine) and attacks were launched through the pivot point.

In the first attack, the compromised machine launches a remote exploit (MS08-netapi in the Metasploit framework) to compromise a second controller. This illustrates an attacker expanding his original foothold.

Two other attacks consist of files being moved through a Metasploit Meterpreter channel. This represents attackers uploading new tools or updating their presence on the machines, by sending a newer version of their malware for example.

Another type is a fingerprinting attack where an attacker characterizes the content of the memory of a controller by sending a succession of read packets. This represents information gathering activity from the attacker.

The final attack type implemented is the sending of an unauthorized command to a controller. The command is a properly formatted WRITE_COIL packet sent to another controller.

Table 1 summarizes the contents of the different packet captures. In cases where malicious activity is included, the ground truth file provided with the data set describes the specific attacks.

5.2. Modbus Covert channel data set

Most of the attacks included in the SCADA network data set are relatively easy to detect. In particular, command and control traffic on rarely used TCP ports could easily be detected by the most naïve intrusion detection systems. In order to provide challenges for advanced intrusion and anomaly detection systems, we also provide data for a Modbus-based command and control channel that uses the least significant bits of Modbus data to carry information. The details of the implementation can be found in Lemay *et al*. [13].

Table 1: Description of the SCADA network data sets

Note: *Manual operations are different in* Run8 *and* Run11

Name	Description	Malicious activity?	Number of entries
Run8	1 hour of regular Modbus traffic including polling and manual operation - 2 MTU, 3 RTU and 10 seconds polling interval	No	72 186
Run11	1 hour of regular Modbus traffic including polling and manual operation - 2 MTU, 3 RTU and 10 seconds polling interval	No	72 498
Run1_6RTU	1 hour of regular Modbus traffic including polling and manual operation - 2 MTU, 6 RTU and 10 seconds polling interval	No	134 690
Run1_12RTU	1 hour of regular Modbus traffic including polling and manual operation - 2 MTU, 12 RTU and 10 seconds polling interval	No	238 360
Run1_3RTU_2s	1 hour of regular Modbus traffic including polling and manual operation - 2 MTU, 3 RTU and 2 seconds polling interval	No	305 932
Modbus_polling_only_6RTU	1 hour of regular Modbus traffic including polling only - 1 MTU, 3 RTU and 10 seconds polling interval	No	58 325
Moving_two_files_Modbus_6RTU	3 minutes of regular Modbus traffic including polling only - 1 MTU, 6 RTU and 10 seconds polling interval	Yes	3 319
Send_a_fake_command_Modbus_6RTU_with_operate	11 minutes of regular Modbus traffic including polling and manual operation - 1 MTU, 6 RTU and 10 seconds polling interval. Also includes sending a Modbus write operation from a compromised RTU using Metasploit proxy functionality and the proxychains tool	Yes	11 166
Characterization_Modbus_6RTU_with_operate	5.5 minutes of regular Modbus traffic including polling and manual operation - 1 MTU, 6 RTU and 10 seconds polling interval Also includes sending a series of modbus read commands to characterize available registers from a compromised RTU	Yes	12 296
CnC_uploading_exe_modbus_6RTU_with_operate	1 minute of regular Modbus traffic including polling and manual operation - 1 MTU, 6 RTU and 10 seconds polling interval. Also includes sending an EXE file from a compromised RTU to another compromised RTU through a Metasploit meterpreter channel	Yes	1 426

6RTU_with_operate	5 minutes of regular Modbus traffic including polling and manual operation - 1 MTU, 6 RTU and 10 seconds polling interval. Also includes using an exploit (ms08_netapi) from a compromised RTU to compromise another RTU using Metasploit	Yes	1 856

The packet captures represent the various configuration of the channel. In particular, they represent the different configurations in terms of number of digits used for storage and the number of Holding Registers used for storage. For example, if the baseline measurement is 12 000 volts, using the two least significant digits would produce a measurement of 12 0XX, where XX is the numerical value of the information transferred. If 3 storage registers are used, the registers of three different measurements will be used for storage.

Theoretically, the more information carried by the covert channel, the easier it should be to detect the channel. However, for this type of analysis, it is important to take into account the information property of the baseline network. In the data described in Section 5.1, there is minimal noise in the physical measurements as the entropy is introduced by random operation of ON/OFF switches. As such, it is possible to easily distinguish the covert channel by looking at the registers with the highest entropy. This might not be applicable to a production system where the measurements are unlikely to be noiseless. Timing issues also have to be considered as the channel does not follow a specific polling frequency and should probably be scaled to match the frequency of the background traffic to correctly evaluate performance.

For the channel capture, only the content of the channel between machines 192.168.43.148 and 192.168.43.149 is provided. The contents represent successive file transfer runs. Also, as only the covert channel traffic is provided, all packets should be considered malicious as per our labeling discipline. Table 2 summarizes the contents of the different packet captures.

Table 2: Description of the covert channel data sets

Name	**Description**	**Malicious activity?**	**Number of entries**
Channel_2d_3s	Modbus covert channel using the two least significant digits of three storage registers	Yes	383 312
Channel_3d_3s	Modbus covert channel using the three least significant digits of three storage registers	Yes	255 668
Channel_4d_1s	Modbus covert channel using the four least significant digits of one storage registers	Yes	414 412
Channel_4d_2s	Modbus covert channel using the four least significant digits of tw0 storage registers	Yes	266 387
Channel_4d_5s	Modbus covert channel using the four least significant digits of five storage registers	Yes	107 577
Channel_4d_9s	Modbus covert channel using the four least significant digits of nine storage registers	Yes	60 295
Channel_4d_12s	Modbus covert channel using the four least significant digits of twelve storage registers	Yes	44 977
Channel_5d_3s	Modbus covert channel using the five least significant digits of three storage registers	Yes	143 809

6. Conclusion

This paper has presented data sets generated in the SCADA sandbox that are publicly available to security researchers. These data sets include different configuration values and provide different attack scenarios with the objective of increasing the reliability of proposed intrusion detection scenarios. Furthermore, the data sets detailed in the paper are accompanied by labels that reveal the ground truth. This makes the data set particularly suitable for use in supervised machine learning approaches.

While the data sets have a number of limitations, for example the lack of fidelity in terms of information properties of the data, lack of variance in controller configuration and uniform polling interval configuration, we feel that the data sets presented offer a significant improvement over the alternatives, or lack thereof.

The public nature of these data sets also enables the community to critique these deficiencies to guide future work in improving the quality of the data sets produced and their verisimilitude with production networks.

Increasing this verisimilitude is the object of future research efforts. Through the acquisition of production data, it would be possible to calibrate the SCADA sandbox to emulate production networks more closely. The addition of a greater variety of attacks would also be an important avenue for future research.

7. Acknowledgments

This research was funded in part by the Canadian Center for Security Science (CSS). In addition, we would like to thank the National Energy Infrastructure Test Center (NEITC) for additional testing and support.

8. References

[1] J. McHugh, "Testing intrusion detection systems: a critique of the 1998 and 1999 DARPA intrusion detection system evaluations as performed by Lincoln Laboratory," *ACM Transactions on Information and System Security,* vol. 3, no. 4, pp. 262-294, 2000.

[2] H. Hadeli, R. Schierholz, M. Braendle and C. Tuduce, "Leveraging determinism in industrial control systems for advanced anomaly detection and reliable security configuration," in *IEEE Conference on Emerging Technologies & Factory Automatio*(ETFA)Mallorca, 2009.

[3] A. Lemay, "Defending the SCADA Network Controlling the Electrical Grid from Advanced Persistent Threats," Ph.D. Thesis, École Polytechnique de Montréal, 2013.

[4] D. Hadžiosmanović, L. Simionato, D. Bolzoni, E. Zambon and S. Etalle, "N-Gram Against the Machine: On the Feasibility of the N-Gram Network Analysis for Binary Protocols," 15th International Symposium on Research in Attacks, Intrusion and Defenses (RAID) . pp. 354–373, Amsterdam, 2012.

[5] R. R. R. Barbosa, A. Pras and R. Sadre, "Flow whitelisting in SCADA networks," in *7th Annual IFIP Working Group 11.10 International Conference on Critical Infrastructure Protection*, Washington, 2013.

[6] R. R. R. Barbosa, R. Sadre and A. Pras, "A First Look into SCADA Network Traffic," in *Network Operations and Management Symposium* (NOMS)Maui, 2012.

[7] A. Valdes and S. Cheung, "Communication Pattern Anomaly Detection in Process Control Systems," in *IEEE Conference on Technologies for Homeland Security*, Waltham, MA, 2009.

[8] W. Gao, T. Morris, B. Reaves and D. Richley, "On SCADA control system command and response injection and intrusion detection," in *2010 eCrime Researchers Summit (eCrime)*, Dallas, 2010.

[9] A. Lemay, J. M. Fernandez and S. Knight, "An isolated virtual cluster for SCADA network security research", in *Proceedings of the 1st International Symposium for ICS & SCADA Cyber Security Research* (ICS-CSR), Leicester, 2013

[10] Sourceforge, "Sourceforge project - SCADA BR," 16 December 2014. [Online]. Available: http://sourceforge.net/projects/scadabr/. [Accessed 26 January 2015].

[11] L. Jean, "Python Software Foundation - modbus_tk 0.4.3," 3 November 2014. [Online]. Available: https://pypi.python.org/pypi/modbus_tk. [Accessed 26 January 2015].

[12] A. Hern, "Ukrainian blackout caused by hackers that attacked media company, researchers say," The Guardian, 7 January 2016. [Online]. Available: http://www.theguardian.com/technology/2016/jan/07/ukrainian-blackout-hackers-attacked-media-company. [Accessed 21 January 2016].

[13] A. Lemay, J. M. Fernandez and S. Knight, "A Modbus command and control channel," in *IEEE Systems Conference* (SYSCON), Orlando, 2016.