Chapter 14

GENERATING ABNORMAL INDUSTRIAL CONTROL NETWORK TRAFFIC FOR INTRUSION DETECTION SYSTEM TESTING

Joo-Yeop Song, Woomyo Lee, Jeong-Han Yun, Hyunjae Park, Sin-Kyu Kim and Young-June Choi

Abstract Industrial control systems are widely used across the critical infrastructure sectors. Anomaly-based intrusion detection is an attractive approach for identifying potential attacks that leverage industrial control systems to target critical infrastructure assets. In order to analyze the performance of an anomaly-based intrusion detection system, extensive testing should be performed by considering variations of specific cyber threat scenarios, including victims, attack timing, traffic volume and transmitted contents. However, due to security concerns and the potential impact on operations, it is very difficult, if not impossible, to collect abnormal network traffic from real-world industrial control systems. This chapter addresses the problem by proposing a method for automatically generating a variety of anomalous test traffic based on cyber threat scenarios related to industrial control systems.

Keywords: Industrial control systems, anomaly detection, traffic generation

1. Introduction

Industrial control systems are used in a variety of critical infrastructure assets such as power plants, waterworks, railways and transportation systems. The security of industrial control systems in the critical infrastructure is a grave concern due to the increased risk of external attacks and the potentially serious impact on operations [7, 23]. Therefore, it is important to develop sophisticated systems that can rapidly and accurately detect anomalous industrial control network behavior due to potential attacks.

Published by Springer Nature Switzerland AG 2018.
J. Staggs and S. Shenoi (Eds.): Critical Infrastructure Protection XII, IFIP AICT 542, pp. 265–281, 2018.
https://doi.org/10.1007/978-3-030-04537-1_14

Intrusion detection systems, which have been used for decades to detect and respond to abnormal operations in information technology systems and networks, are increasingly used in operational technology infrastructures such as industrial control networks. Intrusion detection systems are classified as misuse detection systems and anomaly detection systems [5]. Misuse detection relies on attack signatures – patterns and characteristics – to identify attacks. Therefore, misuse detection is ineffective against zero-day attacks and clever variants of known attacks. In addition, the massive network flows, diversity of attacks and increasing numbers of new attacks make it difficult for modern misuse detection systems to keep up with the threats.

Anomaly detection relies on deviations from normal usage patterns that are specified or learned. The approach is attractive for use in industrial control networks because of their stable structure, predictable traffic and relatively low traffic volumes [1, 20]. An anomaly-based intrusion detection system learns a statistical model of normal activities, which it compares against data pertaining to current activities in order to detect behavioral abnormalities, including those caused by undetected or zero-day attacks.

The same cyber attack can be executed on different targets at different times and with variations in its content. Depending on the environment, an anomaly-based intrusion detection system may or may not detect the same attack. Therefore, to evaluate the performance of an anomaly-based intrusion detection system, extensive testing has to be conducted using variations of each cyber threat scenario, including the targets, attack timing, traffic characteristics and transmitted content. Unfortunately, due to security concerns and the potential operational impact, it is very difficult, if not impossible, to evaluate cyber threat scenarios on real-world industrial control systems.

A solution to this problem is to use a testbed that models a real industrial control network and the physical infrastructure. The testbed can then be employed to collect normal and abnormal traffic. However, a high-fidelity testbed is expensive to implement and operate; in any case, it would never completely model the actual assets. Additionally, it is infeasible to create and analyze a large number of cyber attack scenarios, especially when each scenario can have numerous variations.

Efforts have been made to collect real-world traffic using honeypots [19], but such traffic does not adequately model real industrial control environments. A possible solution is to generate abnormal industrial control network traffic by modifying normal traffic to model cyber threat scenarios while maintaining the characteristics of the normal traffic to the extent possible. For each cyber threat scenario, the nature of anomalous network traffic varies. Therefore, the characteristics of abnormal traffic could be modified based on the specific points of time, target sessions and characteristics of the cyber threat scenarios, and a number of cases could be generated to perform accurate performance analysis. However, depending on the specific scenario, it may be difficult to manually modify normal traffic based on variants of the cyber threat scenario.

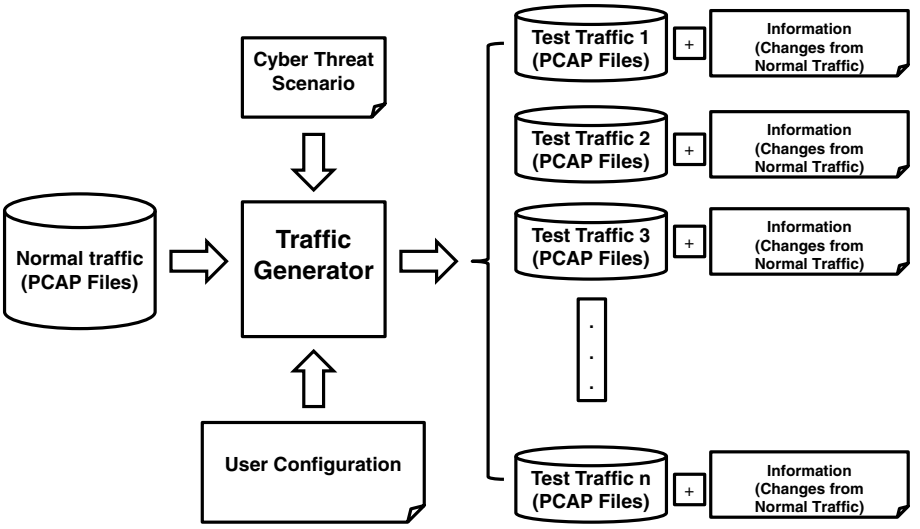

Figure 1. Abnormal test traffic generator.

To address these problems, this chapter proposes a method for automatically generating a variety of anomalous test traffic based on cyber threat scenarios related to industrial control networks. Figure 1 presents a schematic diagram of the abnormal test traffic generator. The automated generation of abnormal traffic requires a method that clearly describes the cyber threat scenarios to be tested. The method involves the specification of "actions" on industrial control network traffic. The characteristics of the point of occurrence, target and abnormal traffic are accordingly adjusted. This creates a number of abnormal scenario cases and abnormal traffic is generated by modifying normal traffic according to each case. Test data can also be generated by combining multiple scenarios.

Packets are the basic communications units of network traffic. In the case of TCP networks, the transmitted data is split into packets, and it is difficult to describe the traffic characteristics by considering individual packets in isolation. On the other hand, in industrial control networks, it is difficult to distinguish transactions since the protocols are often proprietary in nature. Yun et al. [22] have proposed a method for distinguishing transactions in industrial control network traffic. The method, which is shown in Figure 2, distinguishes transactions when the inter-packet arrival time is larger than a predefined threshold. Thus, although the transmitted data is divided into multiple packets, the test traffic is generated in units of transactions that model abnormal traffic more effectively.

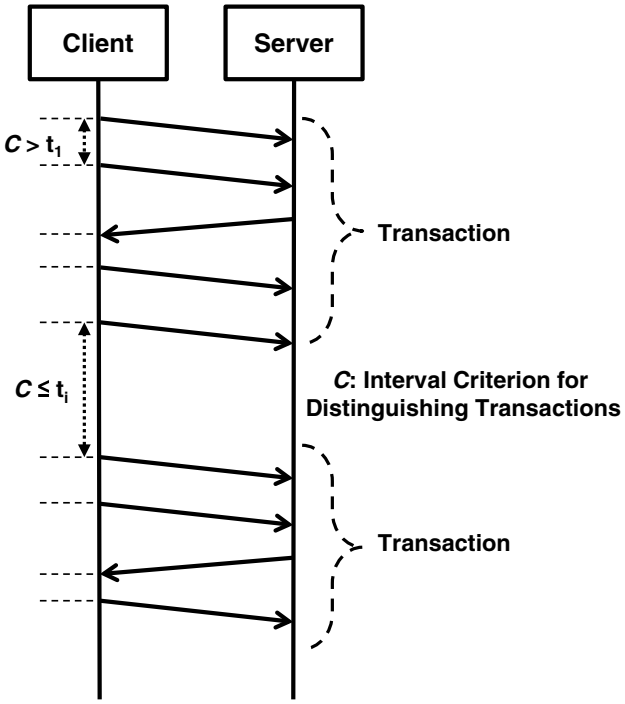

Figure 2. Distinguishing transactions by inter-packet arrival time.

2. Related Work

Some intrusion detection system testing tools generate network traffic that corresponds to known cyber attacks or Snort rules [10, 16, 17]. These tools are useful for measuring the detection rates of intrusion detection systems that rely on attack signatures. However, in order to use these tools for performance analyses of anomaly-based intrusion detection systems, it is necessary to properly mix the generated attack traffic and normal traffic.

Industrial control system testbeds can be used to analyze vulnerabilities, threats and the impacts of attacks. A testbed may be developed using real systems, simulators or a combination of real and simulated systems. Popular simulation tools include Simulink, Stateflow and dSPACE [2, 8, 11]. The tools support automatic code generation, task scheduling and fault management applications for modeling, simulation and testing. Some researchers have used programmable logic controllers and control protocol emulators for constructing honeypots that provide anomalous traffic [3].

SCADA system testbeds have been developed at the national level for security research and analysis. One example is the National SCADA Testbed (NSTB) developed by the U.S. Department of Energy [2]. Other SCADA

testbeds have been evaluated by the U.S. National Institute of Standards and Technology (NIST) and the British Columbia Institute of Technology (BCIT) in Canada. In Europe, testbeds are operational in Grenoble, France; CERN in Geneva, Switzerland; and at the European Joint Research Centre in Ispra, Italy [14]. Christiansson and Luiijf [4] discuss the development of a European SCADA security testbed. Japan also uses an industrial control system testbed for various purposes, including vulnerability analysis [9].

The National SCADA Testbed [2] combines state-of-the-art facilities at national laboratories with expert research, development, analysis and training to identify and address security vulnerabilities and threats in the energy sector. The test and research facilities include field-scale control systems, and advanced visualization and modeling tools.

Other SCADA testbeds have been developed to support similar activities as the National SCADA Testbed. However, they are large and expensive, and are only available to selected researchers. The complexity and scale of a testbed can be reduced, but the results obtained do not adequately model real-world systems. The absence of high-fidelity testbeds that provide open access to researchers has made it very difficult to independently evaluate the research results published in the SCADA systems security literature.

Two other test methods are possible. The first relies on data gathered from real-world systems. In this case, it is possible to perform practical analyses of real traffic. However, it is difficult to conduct evaluations because attack scenarios involve traffic that often does not exist in the captured traffic, requiring attack traffic to be generated artificially.

The second method is to use publicly-available test data provided by organizations that operate testbeds. In the field of industrial control systems, some datasets have been made available, including for secure water treatment [6], S7Comm [18] and Modbus [13]. These datasets enable researchers to quantitatively evaluate the performance of different security techniques and tools. However, when testing anomaly detection systems, it is necessary to experiment with many variations of each abnormal situation. Unfortunately, publicly-available datasets do not maintain adequate amounts of such data.

3. Abnormal Traffic Generation

Given normal traffic and an attack scenario, the test traffic generator (TG) automatically generates a variety of abnormal traffic by changing: (i) target packets (i.e., packets selected to represent anomalies in normal traffic); (ii) generation times (i.e., specific times during which attacks occur repeatedly or regularly in normal traffic); and (iii) applied IP addresses (i.e., changes to the IP source address and/or IP destination address of packets to specific IP addresses corresponding to attack scenarios).

First, the traffic generator selects normal traffic for a certain condition that forms the basis of the scenario. Next, it modifies the selected normal traffic according to the scenario. The basic traffic generation process is as follows:

- **Preprocessing:** The traffic generator receives PCAP-type normal traffic and selects the traffic related to the specific communications section (IP, edge, session and service) specified by the user and uses it to generate abnormal traffic. The selected traffic is referred to as "base traffic."

- **Target Traffic Extraction:** The traffic generator receives the number and length of the target traffic from the user. It then extracts the target traffic by randomly selecting the start time from the base traffic. The target traffic is part of the base traffic and is used to generate abnormal traffic. A variety of abnormal traffic is generated for a single attack scenario by extracting target traffic at various points in time from the base traffic and using it to generate abnormal traffic.

- **Target Traffic Modification:** The traffic generator modifies the target traffic packets according to the attack scenario to generate abnormal traffic. The traffic is transformed by performing an "action" on target traffic. An action involves modifying, adding or deleting some packets or transactions. This creates test traffic corresponding to cyber attacks. The characteristics of the abnormal traffic expected according to the attack are defined as "actions."

  In a real industrial control system, it is highly likely that various types of cyber attacks are performed periodically on multiple devices. To simulate this, n cyber attacks as expressed as n actions, and multiple actions are performed in parallel or sequentially on abnormal traffic. The user selects the number of actions according to the attack scenario and creates a scenario file by selecting elements such as the attack time and frequency, target packet selection criterion and packet transformation method for each action.

  A variety of cases can be created by changing the details of an action based on some condition without fixing it to specific values. In other words, the various test traffic corresponding to an attack scenario can be automatically generated and used for performance evaluation, improving the reliability of the results.

  The traffic generator implements packet-based and transaction-based traffic modifications. A packet is the basic unit of network communications. However, when data is transmitted in the network, it is broken up into multiple packets. For example, when using the TCP protocol, data is divided into several packets and the receiver sends a response to each packet. It is difficult to express the characteristics of such traffic by examining individual packets. An attack is more likely to manifest itself in a transaction than in an individual packet.

3.1 Time and Periodicity of Actions

Multiple actions on target traffic can be performed simultaneously according to each attack cycle. The user has to select the number of actions based on

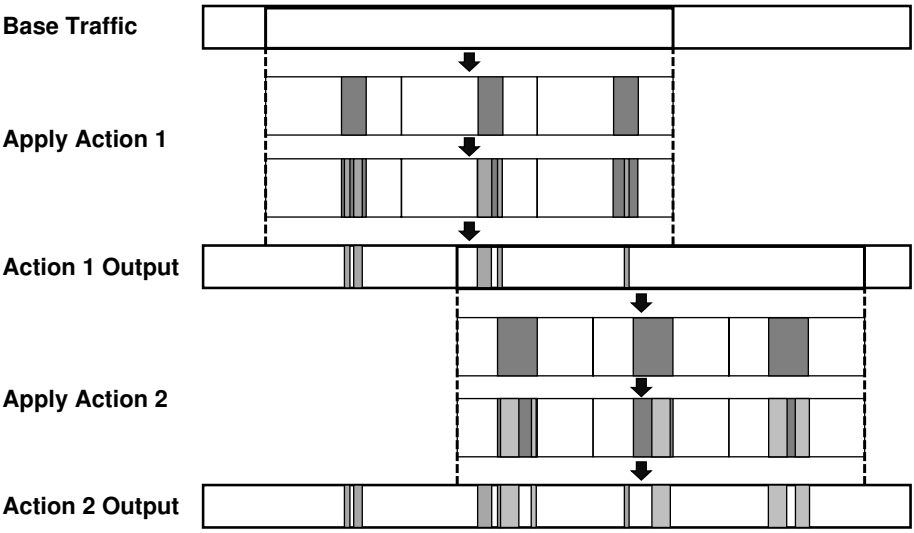

Figure 3. Combination of two actions.

the attack scenario, and then set the attack cycle, attack start time and attack end time for each action.

Figure 3 shows that, when Action 1 is applied, the test traffic, which is the output of Action 1, is generated from the base traffic. When Action 2 is applied to the test traffic transformed by Action 1, test traffic on which Action 1 and Action 2 are simultaneously applied is generated. This procedure generates test traffic corresponding to multiple combined attacks. Since multiple attacks can occur at the same time in a real environment, it is possible to express this situation via multiple action definitions. This can also be used to evaluate whether or not a specific attack type is classified correctly under multiple attacks. If scenarios that define actions are shared and reused, then experimental results and intrusion detection performance can be compared using common actions in normal traffic corresponding to each user. A user can create and test individual traffic with shared actions or traffic with multiple actions in combination with other actions. This produces a variety of anomalous traffic for testing purposes.

3.2 Target Packets of an Action

In order to transform traffic, the target packets used for transformation should be selected for each target data. The target data is part of the target traffic and is segmented at a specific time. A cyber attack on an actual industrial control system involves a specific target IP address, edge (IP address to IP address), session and service. Therefore, the user should specify the criteria for selecting target packets in a scenario.

The traffic generator provides packet-based and transaction-based transformations. When transforming traffic on a transaction-by-transaction basis, the user should select a target transaction instead of a target packet.

The user options for selecting target packets are summarized as follows:

I. Target Types:

1. Attack occurs at a specific IP address.
2. Attack occurs at a specific edge (IP address, IP address).
3. Attack occurs on a specific session (IP address, port, protocol, port, IP address).
4. Attack occurs on a specific service (protocol, port).

II. Number of Target IP Addresses ($\mathbf{N_{TI}}$), Edges ($\mathbf{N_{TE}}$), Sessions ($\mathbf{N_{TSS}}$), Services ($\mathbf{N_{TS}}$):

1. Enter a constant value.
2. Enter an occurrence rate ($x1\%$).

– $N_{Tx1} = x1\%$ of the number of IP addresses/edges/sessions/services used in target traffic.

III. Target IP Address Selection:

1. Input a target IP address/edge/session/service and use it in all the target data.
2. Select a target IP address/edge/session/service randomly for each target data. If a smaller number of IP addresses is used for specific target data, then all the IP addresses in the target data become target IP addresses. The same is true for edge and session.
3. Randomly select target traffic and use it all the target data.

IV. Number of Target Packets ($\mathbf{N_{TP}}$):

1. Enter a constant value.
2. Enter an occurrence rate ($x2\%$).

– N_{TP} = total number of packets in target traffic $\times$ $x2\%$/number of target data.

Based on the four options listed above, the traffic generator selects N_{TP} packets in the target traffic.

3.3 Traffic Modification by an Action

The traffic generator supports four operations for directly modifying target packets or transactions:

- **Payload Change:** Change the payload of the target packet based on the byte section provided by the user.

- **Packet Transmission Rate Change:** Change the packet transmission rate by modifying the packet number of normal traffic by adding or deleting a packet to normal traffic or changing the byte length of the target packet.
- **Packet Replacement/Addition:** Replace the target packet or add an attack packet to the target packet.
- **Header Change:** Change the transmission time and IP address of a target packet. For example, to represent a replay attack, the headers containing the transmission time of the target packet and IP address information should be changed and added to the normal traffic. In order to represent a packet forgery in an intermediate attack on a specific (IP address, IP address) interval, a target packet is selected in the interval and the payload of the selected target packet is modified and added to the normal traffic.

The user options for target traffic transformation are stored in the scenario file. The options are summarized as follows:

I. Payload Change:

1. Change confirmation
   (a) Make a change. When changing to T_R, the same option applies to all the packets in T_R.
   (b) Do not make a change. The remaining options (2 and 3) are not input.
2. Enter a payload change interval (byte).
3. How to change the payload.
   (a) Enter the change value.
   (b) Change to a random value.

II. Packet Transmission Rate Change:

1. Change confirmation.
   (a) Make a change. When changing to T_R, the same option applies to all the packets in T_R.
   (b) Do not make a change. The remaining options (2 and 3) are not input.
2. Count change (increase, decrease or maintain the number of packets).
   (a) Increase: Number of test traffic packets is greater than the number of target traffic packets. Increase the amount of test traffic by copying the target packet based on the packet growth rate ($x3\%$) selected by the user.
   (b) Reduction: Number of test traffic packets is smaller than the number of target traffic packets. Reduce the number of test traffic by deleting part of the target packet based on the packet reduction rate ($x3\%$) selected by the user. $N_{AP} = N_{TP} \times x3\%$.

(c) No change: Number of test traffic packets and number of target traffic packets do not change.

3. Count change (increase, decrease or maintain the number of packets).
   (a) Increase: Change the byte length of the target packet by appending a random value to the end of the target packet.
   (b) Reduction: Remove the payload part of the target packet to change the byte length of the target packet.
   (c) No change.

III. Packet Replacement/Addition:

1. Delete the target packet and replace it with an attack packet.
2. Add an attack packet to the target packet.

IV. Header Change:

1. Change confirmation.
   (a) Make a change. When changing to T_R, the same option applies to all the packets in T_R.
   (b) Do not make a change. The remaining options (2, 3 and 4) are not input.
2. Transmission time change.
   (a) Sequential offset: Transmission time of the target packet is shifted by an offset time provided by the user and employed as the transmission time of the attack packet (at this time, the packet leaving the action period is discarded).
   (b) Sequential random: Keep only the transmission order of the target packet and randomly transmit the generated attack packet in the action period.
   (c) Random: Randomly transmit the generated attack packet in the action period.
   (d) No change.
3. IP address change.
   (a) Randomly change the target packet IP address to an IP address in the base traffic and use it as the IP address of the attack packet.
   (b) Change the target packet IP address to a user-specified IP address.
   (c) No change.
4. Session change.
   (a) Change within the target session.
   (b) Change the session associated with the transmission/reception of the target packet.
   (c) Randomly select one of the (IP address, port) values in the target traffic.
   (d) Change the session to a user-specified session on a transaction-by-transaction basis.
   (e) No change.

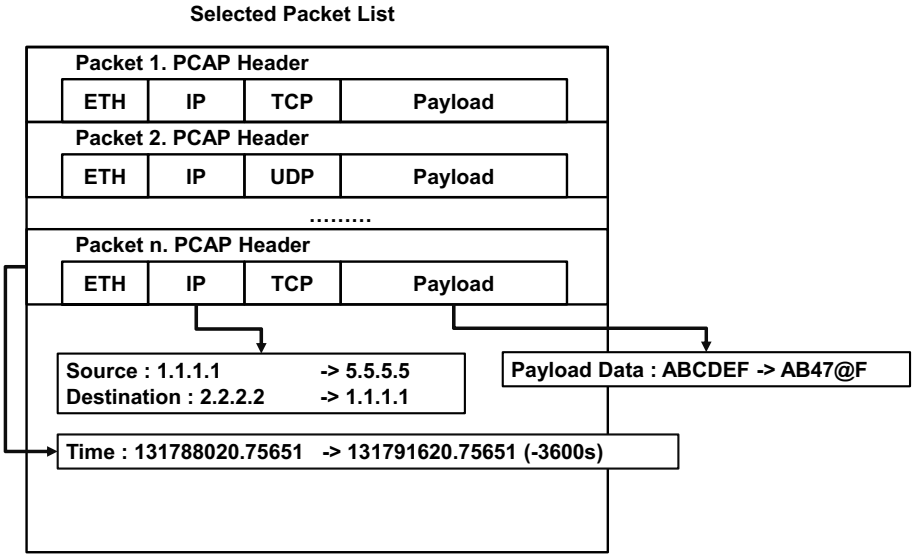

Figure 4. Example of packet modification.

Figure 4 shows an example of packet modification. Packets in the selected list are modified based on the input condition values. The packets are then combined with normal traffic to generate anomalous traffic.

A user may define traffic modifications based on specific cyber threat scenarios by listing the actions that yield the following effects:

- By specifying a protocol, it is possible to express abnormal behavior using a protocol vulnerability or to select abnormal behavior that occurs at a specific point (IP address). By changing the IP addresses in common packets, it is possible to represent a distributed denial-of-service attack that transmits packets from various IP addresses to specific IP addresses, or a man-in-the-middle attack that intercepts packets from certain IP addresses and sends them to other IP addresses.

- Network attacks can occur simultaneously or repeatedly at various time intervals. It is possible to represent attacks that occur at specific times and an attack that occurs repeatedly.

- Increasing the amount of traffic can represent abnormal behavior corresponding to a denial-of-service attack. Reducing the amount of traffic can represent abnormal behavior corresponding to intentional packet drops. Since this method increases or decreases the amount of traffic at several levels, the denial-of-service criterion can be determined by considering the general packet volume and throughput in the network environment. If

throughput information is not available, it is possible to predict a denial-of-service attack by specifying an acceptable scale factor.

- By changing IP addresses, it is possible to express abnormal behavior such as communications at unexpected locations or communications at abnormal times at various locations. Details such as IP addresses, times and transmission content can be created for various cases by changing these configurations in a fixed or random manner or in a specific range. In other words, it is possible to create a large number of cases for a single scenario. The resulting automatically-generated test traffic can support highly-reliable evaluations of intrusion detection system performance.

4. Implementation

In the experiments, PCAP traffic was collected from a real industrial control network and passed to the traffic generator. The traffic collection was accomplished using an application programming interface (API) – `libpcap` for Unix/Linux systems and WinPcap for Windows systems. Since the PCAP traffic was collected in an industrial control network, it contained information about the real environment.

The traffic generator created anomalous PCAP traffic from the collected PCAP traffic, which was added to the original PCAP traffic to create the test PCAP traffic. Since the real environment was reflected in the original traffic, the test traffic captured normal operations as well as attacks. After creating the test traffic, it may be sent to a network, machine learning system or a security device (intrusion detection system or firewall) for learning and testing.

The traffic generator was written in Python 2.7. Wireshark was employed to leverage its PCAP splitting and merging functions (`editcap` and `mergecap`). The `scapy` library was used for PCAP `read` and `write` functions and the `multiprocessing` library was used for speed up. The performance was increased by dividing a large-capacity PCAP file into 1,000 units using `editcap` and then reading it with `multiprocessing`. Note that the selection of 1,000 units was arbitrary and a user may increase or decrease the number of units based on memory availability.

4.1 Preprocessing

The traffic generator receives PCAP-type normal traffic from the collected network traffic and generates base traffic by selecting only the traffic related to specific IP addresses/edges/sessions/services designated by a user. The user inputs a CSV file with preprocessing options to the traffic generator as shown in Table 1. Note that "–" means any and "$r(n)$" means select the number n randomly. If multiple rules (preprocessing conditions) are provided as in Table 1, then the packets that satisfy at least one rule are included in the base traffic.

The following options are included in Table 1:

Table 1. Preprocessing options.

Option	$\mathbf{IP_{src}}$	$\mathbf{Port_{src}}$	Protocol	$\mathbf{IP_{dest}}$	$\mathbf{Port_{dest}}$	Bidirectional
1	IP1	–	–	–	–	No
2	IP2	–	–	IP3	–	No
3	IP4	Port1	–	IP5	Port2	Yes
4	–	–	Proto1	–	–	No
5	IP6	Port3	Proto2	IP7	Port4	Yes
6	$r(50)$	–	–	–	–	No

- **Option 1:** All the packets sent from and received at IP1 (IP address selection).
- **Option 2:** All the packets sent between IP2 and IP3 (edge selection).
- **Option 3:** All the packets sent from IP4-Port1 to IP5-Port2 (session selection).
- **Option 4:** All the packets using Proto1 (service selection).
- **Option 5:** All the packets sent from IP6-Port3 to IP7-Port4 using Proto2.
- **Option 6:** Fifty randomly-selected IP addresses from among the IP addresses in the input data, and all the packets transmitted from and received at the 50 IP addresses.

The traffic generator can also provide information about the IP addresses/edges/sessions/services for traffic that a user can employ to create an attack model. Each file provides a list of IP addresses/edges/sessions/services used by the traffic. If base traffic is already available, the traffic generator can proceed directly to the target traffic generation step without any preliminary work.

4.2 User Configuration File

The traffic generator modifies normal traffic according to the characteristics of an attack scenario to create abnormal traffic. A user inputs a scenario (discussed in Sections 3.2 and 3.3 and Table 1) in the form of a CSV file that embodies the characteristics of the test method and attack scenario. The traffic generator then creates: (i) target traffic according to the options listed in the scenario file; (ii) divides the target traffic into target data representing attack periods; and (iii) generates abnormal traffic by performing actions on the target data. The abnormal traffic that is generated is also in the PCAP format and has the same size as the target traffic.

Table 2 shows a scenario file that simulates a query injection attack by changing the payloads of randomly-selected target packets in target traffic. Since only the payload is changed, not the header, it corresponds to a man-in-the-middle (MiTM) attack. The target traffic is divided into five pieces of

Table 2. Generation of abnormal traffic for a query injection attack.

Target Traffic Generation	- Number of target traffic (N_T): 100 - Length of target traffic(L_T): 5 min		
Target Data Generation	- Number of actions (N_{Act}): 1 - Attack period (P_A): 1 min - Starting point (t_{A1}): 0 min - Ending point (t_{A2}): 5 min (Five target data of one minute in length are generated)		
Action	**Target Packet Type**	- Action period ($t_{A1} \sim t_{A2}$): $0 \sim 60$ s Total period and action period set to same value - Target type: 2. Attack occurs at a specific edge (IP address, IP address) - Number of target IP addresses/edges/sessions/ services: 2. Enter the occurrence rate (x1%) N_{TE} = 1% of number of edges in target traffic - How to specify target edge: 3. Randomly select target traffic and use the same in all target data	
	Target Packet Selection	- Number of target packets (N_{TP}): 2. Enter occurrence rate = 0.01% N_{TP} = Packets in target traffic × 0.0001/5 - Select target packets: 1. Randomly select N_{TP} target packets from packets using target edge in target data	
	Target Traffic Transformation	**Payload**	1. Change confirmation: (a) Perform the change 2. Enter payload change interval: 1∼5 bytes 3. How to change payload: (c) Change to random value
		Traffic Volume	1. Change confirmation: (b) No change
		Replacement/ Addition	1. Delete target packet and replace it with attack packet
		Header	1. Change confirmation: (b) No change

target data of one minute each to perform an action. If the action period and total period are the same, then the target data length would be meaningless because the attack does not have any periodicity.

When it is executed, the traffic generator produces the target packet list, modified packet list, test traffic and test traffic log information. By comparing the target packet list against the modified packet list, it is possible to con-

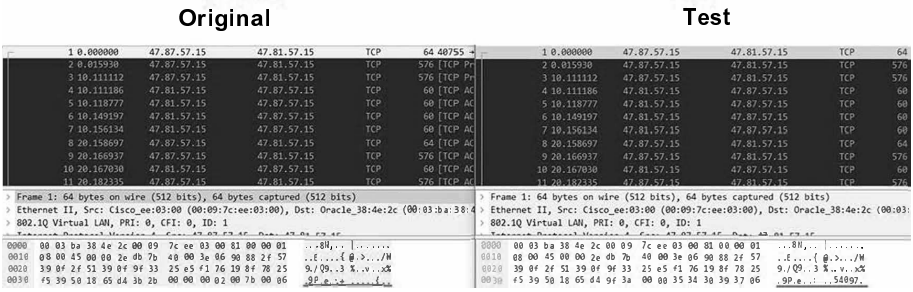

Figure 5. Traffic generation for a query injection attack.

firm whether or not traffic modifications were performed based on the attack scenario. The test traffic log stores the options pertaining to the scenario file and information about traffic generation. After the test traffic is generated using the scenario file and the collected industrial control network traffic, the resulting target traffic and modified packets are shown in Figure 5.

The proposed approach can change protocol commands as desired (e.g., to DNP3, IEC61850 or Modbus). Injection is modeled by specifying the byte portion that contains the command and changing it to another command desired by the user. This method handles bytes; therefore, if the structure of the protocol is known, the desired protocol commands can be generated.

5. Conclusions

The principal challenge in conducting research on securing industrial control networks from cyber attacks is the lack of availability of real-world network traffic that reflects normal and anomalous operations. Although it is possible to collect traffic under normal operating conditions, due to security concerns and the potential impact on operations, it is very difficult, if not impossible, to collect abnormal network traffic from real-world industrial control systems. While testbeds can overcome this limitation, they are expensive to implement and operate; moreover, they will never completely model their real counterparts. Additionally, it is infeasible to create and analyze a large number of cyber attack scenarios, especially when each scenario can have numerous variations.

This chapter has addressed the problem by proposing a method for automatically generating a variety of anomalous test traffic based on cyber threat scenarios related to industrial control systems. The proposed method starts with normal traffic that is collected from a real industrial control network. Leveraging abnormal scenarios provided by users, the method automatically generates anomalous (attack) traffic based on target connections, time, traffic amounts and transmission content that satisfy the scenarios. The anomalous traffic is added to the original traffic to create the test traffic for developing and evaluating intrusion detection systems.

Future research will enhance the automated traffic generation process to capture novel and multistage attacks. Additionally, it will attempt to model the potential impacts of traffic with manipulated packets and/or transactions on real industrial control devices.

References

[1] R. Barbosa, R. Sadre and A. Pras, A first look into SCADA network traffic, *Proceedings of the IEEE Network Operations and Management Symposium*, pp. 518–521, 2012.

[2] H. Christiansson and E. Luiijf, Creating a European SCADA security testbed, in *Critical Infrastructure Protection*, E. Goetz and S. Shenoi (Eds.), Springer, Boston, Massachusetts, pp. 237–247, 2007.

[3] Conpot Development Team, CONPOT: ICS/SCADA Honeypot (`conpot.org`), 2018.

[4] C. Davis, J. Tate, H. Okhravi, C. Grier, T. Overbye and D. Nicol, SCADA cyber security testbed development, *Proceedings of the Thirty-Eighth North American Power Symposium*, pp. 483–488, 2006.

[5] O. Depren, M. Topallar, E. Anarim and M. Ciliz, An intelligent intrusion detection system (IDS) for anomaly and misuse detection in computer networks, *Expert Systems with Applications*, vol. 29(4), pp. 713–722, 2005.

[6] J. Goh, S. Adepu, K. Junejo and A. Mathur, A dataset to support research in the design of secure water treatment systems, *Proceedings of the Eleventh International Conference on Critical Information Infrastructures Security*, pp. 88–99, 2016.

[7] A. Hahn and M. Govindarasu, Cyber attack exposure evaluation framework for the smart grid, *IEEE Transactions on Smart Grid*, vol. 2(4), pp. 835–843, 2011.

[8] A. Hahn, B. Kregel, M. Govindarasu, J. Fitzpatrick, R. Adnan, S. Sridhar and M. Higdon, Development of the PowerCyber SCADA security testbed, *Proceedings of the Sixth Annual Workshop on Cyber Security and Information Intelligence Research*, article no. 21, 2010.

[9] Information-Technology Promotion Agency, About IPA, Tokyo, Japan (`www.ipa.go.jp`), 2018.

[10] Ixia, Test Architecture, Calabasas, California (`www.ixiacom.com/solutions/test-architecture`), 2018.

[11] M. Knauff, J. McLaughlin, C. Dafis, D. Niebur, P. Singh, H. Kwatny and C. Nwankpa, Simulink model of a lithium-ion battery for the hybrid power system testbed, *Proceedings of the ASNE Intelligent Ships Symposium*, 2007.

[12] A. Lazarevic, L. Ertoz, V. Kumar, A. Ozgur and J. Srivastava, A comparative study of anomaly detection schemes in network intrusion detection, *Proceedings of the SIAM International Conference on Data Mining*, pp. 25–36, 2003.

[13] A. Lemay and J. Fernandez, Providing SCADA network datasets for intrusion detection research, *Proceedings of the Ninth USENIX Workshop on Cyber Security Experimentation and Test*, 2016.

[14] S. Luders, Control systems under attack? *Proceedings of the Tenth International Conference on Accelerator and Large Experimental Physics Control Systems*, 2005.

[15] S. Mukkamala, G. Janoski and A. Sung, Intrusion detection using neural networks and support vector machines, *Proceedings of the International Joint Conference on Neural Networks*, vol. 2, pp. 1702–1707, 2002.

[16] pevma, rule2alert (`github.com/pevma/rule2alert`), 2014.

[17] pytbull, What is pytbull? (`pytbull.sourceforge.net`), 2018.

[18] N. Rodofile, T. Schmidt, S. Sherry, C. Djamaludin, K. Radke and E. Foo, Process control cyber attacks and labeled datasets on S7Comm critical infrastructure, *Proceedings of the Twenty-Second Australasian Conference on Information Security and Privacy*, Part II, pp. 452–459, 2017.

[19] J. Song, H. Takakura, Y. Okabe, M. Eto, D. Inoue and K. Nakao, Statistical analysis of honeypot data and building of Kyoto 2006+ dataset for NIDS evaluation, *Proceedings of the First Workshop on Building Analysis Datasets and Gathering Experience Returns for Security*, pp. 29–36, 2011.

[20] K. Stouffer, V. Pillitteri, S. Lightman, M. Abrams and A. Hahn, Guide to Industrial Control Systems (ICS) Security, NIST Special Publication 800-82, Revision 2, National Institute of Standards and Technology, Gaithersburg, Maryland, 2015.

[21] J. Wan, A. Canedo and M. Al Faruque, Security-aware functional modeling of cyber-physical systems, *Proceedings of the Twentieth IEEE Conference on Emerging Technologies and Factory Automation*, 2015.

[22] J. Yun, S. Jeon, K. Kim and W. Kim, Burst-based anomaly detection for the DNP3 protocol, *International Journal of Control and Automation*, vol. 6(2), pp. 313–324, 2013.

[23] B. Zhu, A. Joseph and S. Sastry, A taxonomy of cyber attacks on SCADA systems, *Proceedings of the International Conference on Internet of Things and Fourth IEEE International Conference on Cyber, Physical and Social Computing*, pp. 380–388, 2011.