ISSN 0146-4116, Automatic Control and Computer Sciences, 2016, Vol. 50, No. 1, pp. 54–62. © Allerton Press, Inc., 2016.

A SCADA Testbed for Investigating Cyber Security Vulnerabilities in Critical Infrastructures[1]

Abebe Tesfahun and D. Lalitha Bhaskari

AUCE(A), Andhra University, Visakhapatnam-530003, A.P., India

e-mail: abesummit@yahoo.com, lalithabhaskari@yahoo.co.in

Received August 3, 2015; in final form, October 16, 2015

Abstract—Supervisory Control and Data Acquisition (SCADA) systems are widely used in critical infrastructures such as water distribution networks, electricity generation and distribution plants, oil refineries, nuclear plants, and public transportation systems. However, the increased use of standard protocols and interconnectivity has exposed SCADA systems for potential cyber-attacks. In recent years, the cyber-security of SCADA systems has become a hot issue for governments, industrial sectors and academic community. Recently some security solutions have been proposed to secure SCADA systems. However, due to the critical nature of SCADA systems, evaluation of such proposed solutions on real system is im-practical. In this paper, we proposed an easily scalable and reconfigurable virtual SCADA security testbed, which can be used for developing and evaluating SCADA specific security solutions. With Distributed Denial of Service (DDoS) and false data injection attack scenarios, we demonstrated how attackers could disrupt the normal operation of SCADA systems. Experimental results show that, the pro-posed testbed can be effectively used for cyber security assessment and vulner-ability investigation on SCADA systems. One of the outcomes of this work is a labeled dataset, which can be used by researchers in the area of SCADA security.

Keywords: testbed, SCADA, modbus, cyber-attack, critical infrastructure

DOI: 10.3103/S0146411616010090

1. INTRODUCTION

Supervisory Control and Data Acquisition (SCADA) systems are widely distributed systems used to continuously supervise and monitor critical infrastructures such as water distribution networks, electricity generation and distribution plants, oil refineries, nuclear plants, and public transportation systems [1]. By automating data collection from field devices in remote sites, SCADA systems provide operators with a real-time view of the whole process. SCADA systems and the processes they manage are critical to any nation's defense and economy.

For the purpose of reducing costs and increasing efficiency, the SCADA technology migrated from isolated (monolithic) systems to networked architectures that communicate with corporate network and the internet. This enabled the replacement of proprietary protocols by open SCADA protocols; it also allowed interoperability between different Remote Terminal Unit (RTU) vendors. However, this improvement has been implemented on top of most existing SCADA systems without considering its impact towards cyber-security [2, 3]. Security breaches against SCADA systems can disrupt and damage the operation of critical infrastructures, contaminate the ecological environment, cause huge economic losses and even more dangerously human lives loss [4]. There have been several documented incidents and cyber-attacks targeting SCADA systems in different parts of the world. Siberian pipeline explosion is believed to be the first known cyber-security incident against SCADA systems. In 1982, an attacker added a Trojan Horse in the SCADA system that controls the Siberian pipeline, and eventually led to an explosion of 3 kilotons of explosive material [5]. Similarly, in 1999 an attacker collaborated with an insider used Trojan Horse to command the central switchboard which controls gas flow of Gazprom company [5]. In March 2000, in Australia a former contractor at Maroochy water services hacked the system. The pumps were not working properly and alarms were not giving alert [6]. In 2003, the Davis-Besse nuclear plant in Oak Harbor, Ohio, was attacked by Slammer Worm. The worm made the safety parameter display system of the plant down for almost five hours [5].

[1] The article is published in the original.

In June 2010, Stuxnet worm, the new cyber-weapon of cyber-warfare, had struck the Iranian nuclear facility. The Stuxnet worm is the first malware to specifically target industrial programmable logic control systems. The worm uses Siemens' default passwords and gradually modifies the code running in Programmable Logic Controllers to behave them different from their primary purpose. More specifically, Stuxnet alters the frequency of electric current; as result, the drives switch between high and low speeds for which they were not designed [5, 7–9].

After all those incidents, SCADA security has become a concern for researchers, industries and governments. In the past few years, a lot of effort has been made to improve the security of SCADA system. Access control, authentication and intrusion detection are some of the most commonly used security mechanisms to secure critical infrastructures. Since most of the critical infrastructures with SCADA system operate 24/7 and require huge investment, it is not practical to make research on actual SCADA system. Furthermore, for the reason of confidentiality, it is very challenging to collect network traffic signatures on real CADA system for evaluating various SCADA specific security solutions. To address these problems, it would be feasible to develop SCADA testbed for critical infrastructures security research. In the subsequent paragraphs of this section, we will summarize some of the existing SCADA testbeds.

Some countries established government sponsored SCADA testbeds at national level. To facilitate security research and development in the energy sector the National SCADA Test Bed (NSTB) has been established since 2003 by the U.S. Department of Energy [10]. Similarly, in Japan Control System Security Centre Tohoku Tagajo Headquarter Testbed has been established since 2012 [11].

For academic and research purpose Morris et al. [12] proposed a small-scale real SCADA testbed for various industrial applications. On the other hand, Davis et al. [13] proposed simulation based SCADA testbed for power grid. They used network client, PowerWorld Server and RINSE network emulator to represent the control station, power station and communication networks respectively. Chabukswar et al. [14] used C2WindTunnel to simulate the controllers and DDoS attacks on a well-known chemical processing system called Tennessee Eastman Control Challenge.

Kush et al. [15] proposed a smart grid testbed. They used Common Open Research Emulator (CORE) platform to emulate the communication network of the smart grid. They also used C++ to implement the Modbus protocol. The testbed is evaluated against set of smart grid testbed requirements. However, as stated by the authors, the result from security perspective is very anecdotal.

Besides these testbeds, several virtual or simulation based SCADA testbeds have also been proposed in recent years [16–19]. However, most of these testbeds are not freely available. Even those testbeds, which are developed on open software, are sector and application specific. Moreover, these testbeds lack re-configurability for different attack scenarios. In this paper, to address these limitations, an open SCADA testbed is proposed, which can be easily extended to various critical infrastructure sectors. More importantly, different from previously developed testbeds, the proposed SCADA testbed is user friendly and easily reconfigurable for different types of cyber-attacks. In addition to this, in this testbed multiple attacks can be simultaneously initiated from different places in the testbed. This feature of the testbed helps security researches to investigate the effect of simultaneous attacks on SCADA systems. The proposed SCADA testbed uses Common Open Research Emulator (CORE) to simulate SCADA networks. Modbus TCP/IP protocol is used for communication between the SCADA components. The testbed can be used for cyber security assessment and vulnerability investigation on SCADA systems. With our testbed, one can also generate benchmark dataset to develop and evaluate attack detection and protection technologies for SCADA systems. With the current version of the testbed, we have already collected network and control data for SCADA security research and development. Though the testbed is primarily developed for SCADA security research, it can also be used for educational purpose.

2. GENERAL SCADA SYSTEM ARCHITECTURE AND PROTOCOLS

Though the implementation of SCADA systems can largely vary, a generic SCADA system architecture may be obtained by considering the main building components of SCADA system. A typical SCADA system consists of Field Station, Control Station and Communication Network. A typical SCADA system is shown in Fig. 1.

The field station includes physical processes being controlled by SCADA system, instrumentation devices—such as sensors and actuators—and Remote Terminal Units (RTU). These devices are located in the remote site. The instrumentation devices are directly connected to the physical equipment—such as, valves and pumps. The dynamics of the physical system is measured by sensors, and based on measurement values appropriate actions are taken by actuators under the supervision and control of the controller in the control station. The RTUs provide communication interface to the instrumentation devices. The

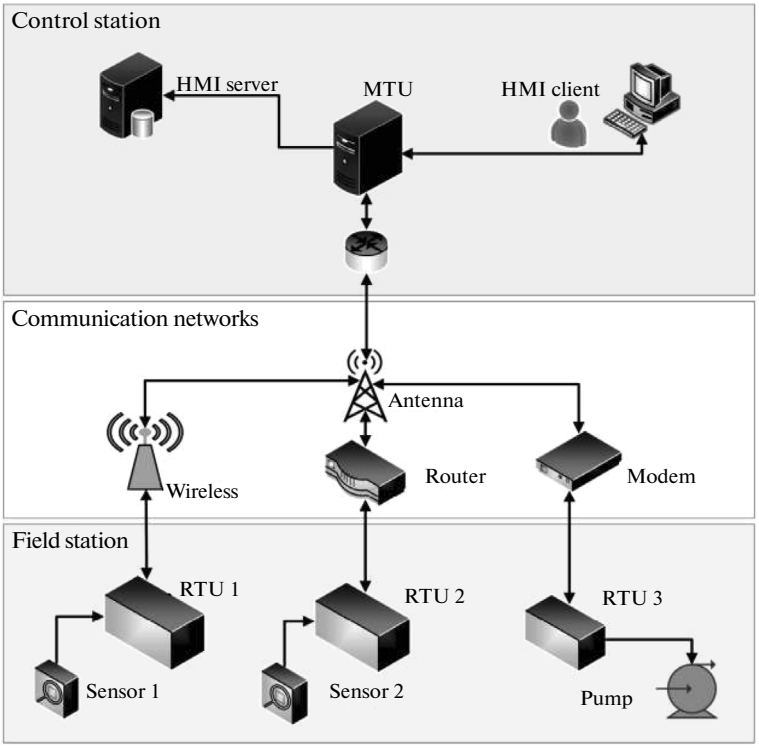

Fig. 1. A typical SCADA system architecture.

RTUs convert electrical signal from sensors to digital signal for communicating with the control station. They also convert digital signals (commands) coming from the control station into electrical analog signal for controlling field devices such as actuators. In some critical infrastructures, the role of the RTU is played by Programmable Logic Controllers (PLCs) and Intelligent Electronic Devices (IEDs).

The control station includes Mater Terminal Unit (MTU) and Human Machine Interface (HMI). The MTU is the heart of the SCADA system that supervises and controls the activities of various RTUs and sends the control commands to the physical process. The MTU uses polling technique to collect data from RTU. Based on the collected data the HMI provides a visual representation of the remote process for operators. The SCADA system through its HMI also provides flexibility for operators to change plant configurations.

The measurement data collected by sensors will be forwarded to the control layer through the communication network. On the other hand, the commands coming from the control station will be sent to actuators through the communication networks. SCADA systems use various communication media including telephone, fiber optic, radio, and satellites to connect the remote terminal units with the control station. For efficient data transfer between the RTU and MTU over a communication network the data has to be encapsulated within a SCADA protocol. Nowadays there are many types of SCADA communication protocols. Modbus TCP [20, 21], Distributed Network Protocol version 3 (DNP3) and Profibus are some of the most widely used protocols.

In this work Modbus protocol is used for communication between the components of the proposed SCADA testbed. Modbus protocol supports request-response messaging between master and slave. Modbus RTU and Modbus TCP/IP are the two variants of Modbus protocol. In this study the Modbus TCP/IP version of Modbus protocol is considered. In Modbus TCP/IP, the Modbus packet is embedded in the TCP segment and 502 TCP port is dedicated to the protocol.

3. PROPOSED SCADA SECURITY TESTBED

The proposed virtual SCADA testbed is developed on Common Open Research Emulator (CORE) in a virtual Linux machine. Each components of the SCADA system are implemented using Python script-

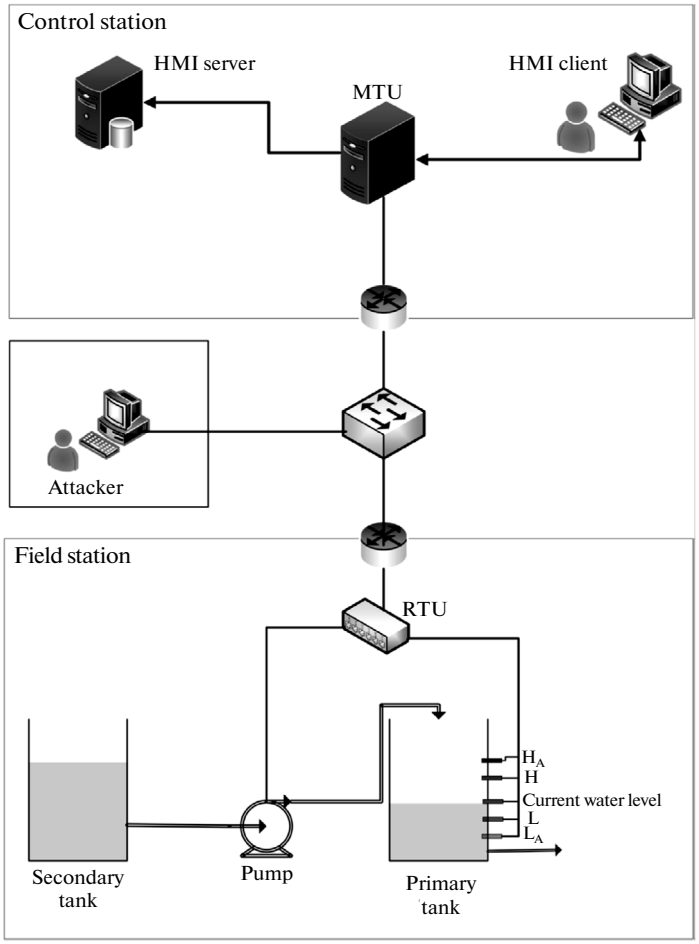

Fig. 2. Proposed SCADA testbed architecture for a simple water tank system.

ing language. The architecture of the proposed SCADA testbed for a simple water tank system is shown in Fig. 2.

Simulation Platform

The proposed testbed is developed on Common Open Research Emulator (CORE) [22]. CORE has been developed by Network Technology research group in Boeing Research and Technology division. CORE framework uses light weight virtualization on Linux systems. On Linux operating system, a virtual node built on CORE framework uses Linux network namespace and the virtual nodes are connected to the virtual networks using Linux Ethernet bridging. CORE emulator is flexible, scalable, and has easy to use GUI. Moreover, CORE emulator can be easily customized and connected to real world networks. The Python module of CORE emulator enables users to customize their own script to emulate any network.

Only some protocols are built in the current version of CORE emulator. SCADA specific communication protocols are not included in CORE emulator. Almalawi et al. [16] proposed a mechanism to integrate Modbus in CORE emulator. They integrated Modbus as service in the emulator. Since in SCADA system there might be large number of RTUs or PLCs with different functionalities it is not feasible to include all such component as service in CORE emulator. In our proposed approach, we developed a python script for each of the components of the SCADA system based on their functionalities. In CORE, we represented each of the SCADA devices by a virtual node, and they are connected to each other by hubs in Local Area Networks (LAN).

Table 1. Process description for automatic operation of the water tank system

Input (sensors)				Output (actuators)	
L_A	L	H	H_A	Pump	Alarm
1	0	0	0	1	1
1	1	0	0	1	0
1	1	1	0	0	0
1	1	1	1	0	1

Communication Protocol

Nowadays there are many types of SCADA communication protocols. Modbus, Profibus and DNP3 are some of the most widely used protocols. For the proposed system, Modbus TCP/IP is used as a communication protocol between the components of the testbed. The Pymodbus library [23] implementation of Modbus TCP/IP is used in the proposed SCADA testbed.

Component Simulation

The proposed SCADA testbed has the following virtual device—Instrumentation devices (sensors and pump), RTU, MTU and HMI. Using socket connection the field devices communicates with RTU by sending and receiving messages. The sensors send message to the RTU about the current status of the plant, and actuators receive control signal from the RTU.

In this testbed, the RTUs act as slave-servers and the MTUs as master-clients. Communication can only be initiated by the master-client. Hence the RTU can only operate upon request of the MTU. RTUs receive commands from the MTU and send back responses to the MTU. The MTU is the heart of a SCADA system. The MTU receives commands and plant parameter setting from the user or operator through HMI. It also sends message to HMI server regarding the current status of the plant. Similarly, based on the current status of the field devices and the parameter settings provided by the operator, the MTU sends message to the RTU.

In the proposed system, the HMI has two variants: HMI server and HMI client. The HMI client requests the HMI server about the current status of the plant and then gives this information to the operator in the graphical form. The operator can change the parameter settings of the plant on HMI client. The HMI client forwards these values to the MTU. Based on these values, valid decisions are taken by the MTU.

Process Simulation

By simply modifying the python code of the process module, the proposed testbed can easily be extended to various types of cyber physical systems. However, for the purpose of simplicity, the critical infrastructure considered in this study is a water tank system. Since liquid processing is common in most critical infrastructure, the proposed SCADA testbed for the water plant can easily be adapted to most fluid processing plants. The water storage tank control system in [12] type is adapted in this paper. In this system two water tanks—primary and secondary tanks, a water pump, five water level sensors and an alarm are used as shown in Fig. 2. The purpose of the pump is to fill water from the secondary tank to primary storage tank. The four water level sensors are fixed at height L_A, L, H and H_A ($L_A < L < H < H_A$) of the primary tank. In addition, the other sensor reads current water level.

The water storage tank system can operate in both automatic and manual modes. In the automatic mode, under normal operation condition the level of the water in the primary tank should oscillate between L and H. It is the responsibility of the MTU to control the level of the water by making the pump ON and OFF. If the water level is at L, the pump should be ON. If the water level reaches H, the pump will be OFF. If, accidentally or by some means, the water level reaches L_A alarm should be ON for the operator to indicate that the tank is going to be empty. Similarly, if the water level reaches H_A alarm should also be ON to indicate that the tank is almost full. To make the process more clear, the automatic operation of the system is represented in Table 1. In a manual mode the operator can make the pump ON or OFF manually. Similar to the automatic mode, in the manual mode the alarm will be triggered if the water level reaches L_A or H_A.

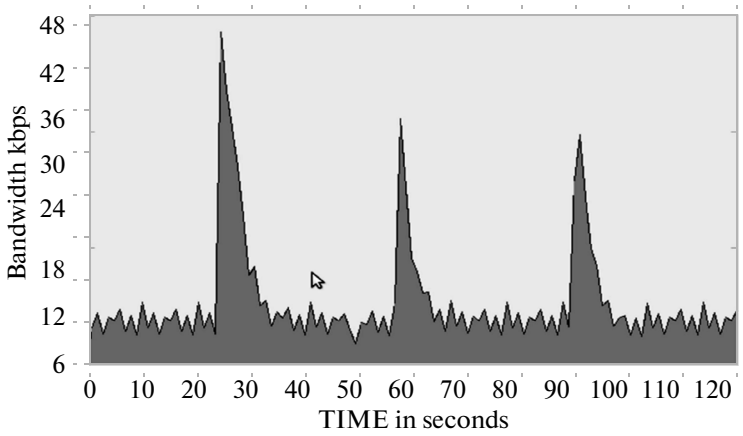

Fig. 3. The effect of DDoS attack on RTU with the maximum bandwidth of the system set to 50 kbps.

In real world water distribution plant there are large numbers of water tanks and pumps. For the sake of simplicity only one pump and five water level sensors are considered as field devices in the proposed SCADA testbed. Hence, one RTU with five registers that store the values of L_A, L, H, H_A and current water level in the primary tank is enough to implement the field station of the plant under consideration. The RTU has also four coils to store binary values of Mode, manual Pump ON/OFF setting, current pump status and current alarm status.

All the simulation components and the process are implemented using python scripting on top of CORE framework. In the next section we present different attack scenarios and their effects in SCADA system.

4. SCADA CYBER-ATTACK SCENARIOS AND RESULTS

Before discussing the attack scenarios, for the sake of result comparison let us divide the vertical length of the primary tank into 10 equal levels. Mark the bottom and upper part of the tank as 0 and 10 respectively. As shown in Fig. 2, under normal condition let $L_A = 2$, $L = 3$, $H = 7$ and $H_A = 8$. In the proposed system Distributed Denial-of-Service (DDoS) and false data injection attacks are considered.

DDoS Attack

To observe the effect of DDoS attack on the proposed testbed we wrote a simple DDoS attack script using python. When this script runs on the attacker node, the RTU will be flooded with packets on port 502. In CORE emulator the maximum bandwidth of the link between two network nodes can be easily configured by a user. For this specific experiment we set the maximum bandwidth of the system to 50 kbps. By flooding the RTU with packets, the bandwidth of the link between the MTU and RTU can be consumed. The more the number of packets coming from the attacker the higher the bandwidth consumed. At some point all the allowed bandwidth will be consumed by the attacker. In this situation the RTU will not listen requests or commands from the MTU. In such scenarios the normal operation of the SCADA system will be disrupted. Figure 3 shows the bandwidth consumed on the link between the RTU and MTU. The experiment was made for 120 s. In this period on the attacker machine we made DDoS attack three times on the RTU. The first attack was attempted between 20 to 30 s and the second and the third were from 55 to 65 and 90 to 100 s respectively. As it can be seen from Fig. 3, in those periods the attacker easily consumed the bandwidth of the link between the RTU and MTU.

False Data Injection Attack

In such type of attacks the attacker manipulates command and control messages. To make false data injection attack on SCADA system, prior knowledge of the process and components of the SCADA system is necessary. In the proposed system, attacks like man-in-the-middle (MITM) are attempted by intercepting command and control messages between the MTU and RTU. We have used a man-in-the-middle attack-generating tool called Ettercap [24]. Ettercap is free powerful tool for MITM attack. This tool has

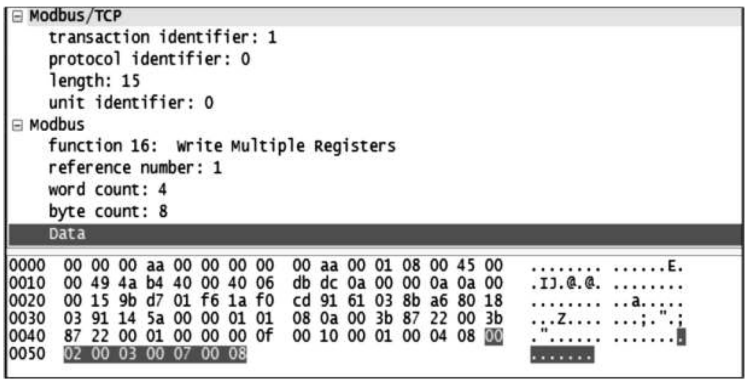

Fig. 4. A command packet captured by Wireshark at the RTU in normal operation of the testbed.

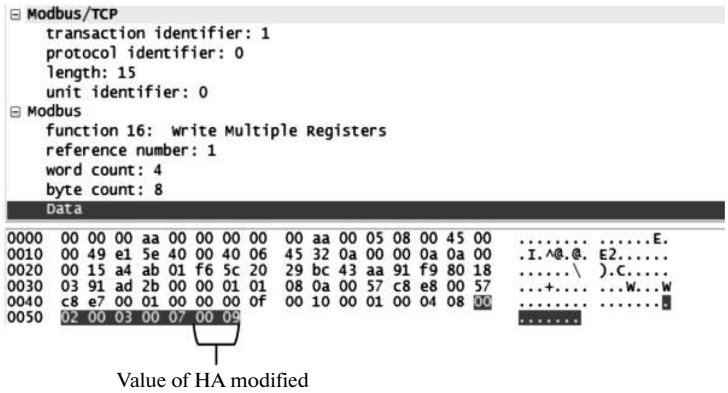

Fig. 5. A command message captured at the RTU after false data injected on a message.

the capability of dissecting many protocols. In order to simulate false data injection attack on a Modbus packet, the command messages from the MTU to RTU and the response messages from the RTU to the MTU are intercepted and modified.

Figure 4 shows a command packet captured by Wireshark at the RTU in normal operation of the testbed. In the normal condition, the data part of Modbus TCP command message is 0x0002, 0x0003, 0x0007, 0x0008, which represents the values of L_A, L, H, and H_A respectively. For the same command message, the packet captured at the RTU after false data injected on a message by the attacker is shown in Fig. 5. Now data part of Modbus TCP command message is changed to 0x0002, 0x0003, 0x0007, and 0x0009. The attacker intercepted the packet and modified the value of HA from 0x0008 to 0x0009, hence alarm will not be ON when $H_A = 8$.

Similarly, the attacker can modify any command message as required. Figure 6 shows a response packet captured by Wireshark at the MTU in normal operation of the testbed. The data part of this packet indicates (0x0002) the current water level at the primary tank. In false response data injection attack scenario, on-the-fly the attacker modifies the packet sent to MTU from RTU. For the same response message, the packet captured at the MTU after false data injected on a message by the attacker is shown in Fig. 7. The attacker easily changed the actual current water level to some other value (0x0007). Though the actual current water level is 2 the MTU and the operator at the HMI client see the value 7 as a true value of the current water level, and they may wrongly decide to stop the pump. This may make the tank to be empty.

5. CONCLUSION AND FUTURE WORK

In this paper, we presented a simple but scalable SCADA security testbed, which contains SCADA components that mimic real SCADA components. The developed testbed is user friendly and easily reconfigurable for different types of cyber-attacks. In the testbed, simultaneous attacks can be initiated

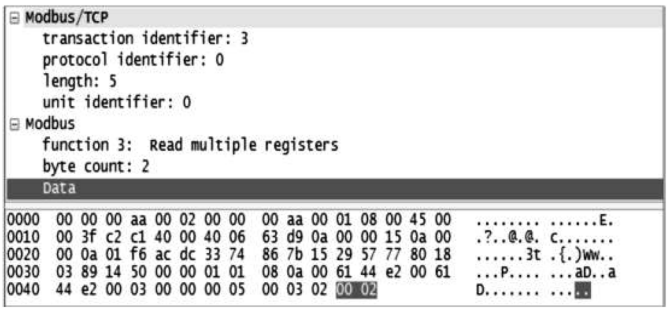

Fig. 6. A response packet captured by Wireshark at the MTU in normal operation of the Testbed.

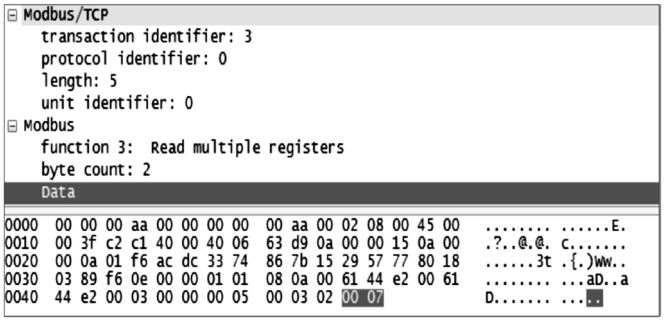

Fig. 7. A packet captured at the MTU after false data injected on a message by the attacker.

from different nodes in the testbed. This feature of the testbed helps security researches to investigate the effect of simultaneous attacks on SCADA systems. The testbed was developed on top of CORE emulator. In the proposed testbed Modbus TCP/IP is used as a communication protocol. With DDoS and false data injection attack scenarios, we demonstrated how attackers could disrupt the normal operation of SCADA systems. With the current version of the testbed, we have already collected network and control data for SCADA security research and development. The collected dataset will be freely available for security researchers in a near future. From results, we can conclude that the developed testbed can be effectively used for cyber security assessment and vulnerability investigation on SCADA systems.

In the future, we would like to evaluate the testbed with several SCADA communication protocols. Furthermore, we will setup an arrangement to make the testbed to be remotely accessible for researchers in the area of SCADA security. Using the data collected from the testbed, we also plan to develop and evaluate a new SCADA specific intrusion detection system.

REFERENCES

1. Stouffer, K., Falco, J., and Scarfone, K., *Guide to Industrial Control Systems (ICS) Security: Supervisory Control and Data Acquisition (SCADA) Systems, Distributed Control Systems (DCS), and Other Control System Configurations such as Programmable Logic Controllers (PLC),* Gaithersburg, MD: National Institute of Standards and Technology, 2011, Spec. Publ. 800-82.
2. Hansen, S., An intrusion detection system for supervisory control and data acquisition systems, *Master's Thesis,* Queensland University of Technology, Brisbane, 2008.
3. Krutz, R.L., Securing SCADA Systems, Indianapolis, IN: Wiley Pub, 2006.
4. *Critical Infrastructure Protection: Multiple Efforts to Secure Control Systems are under Way, but Challenges Remain,* Washington, DC, 2007.
5. Miller, B. and Rowe, D., A survey of SCADA and critical infrastructure incidents, *Proceedings of the 1st Annual Conference on Research in Information Technology,* 2012, pp. 51–56.
6. Slay, J. and Miller, M., Lessons learned from the Maroochy water breach, in *Critical Infrastructure Protection,* Springer, 2008, vol. 253, pp. 73–82.
7. Karnouskos, S., Stuxnet worm impact on industrial cyber-physical system security, *37th Annual Conference on IEEE Industrial Electronics Society,* 2011, pp. 4490–4494.

8. Langner, R., Stuxnet: Dissecting a cyberwarfare weapon, *IEEE Secur. Priv. Mag.,* 2011, vol. 9, no. 3, pp. 49–51.
9. Chen, T.M. and Abu-Nimeh, S., Lessons from Stuxnet, *Computer,* 2011, vol. 44, no. 4, pp. 91–93.
10. DOE National SCADA Test Bed Program Multi-Year Plan. US DOE Office of Electricity Delivery and Energy Reliability, 2008.
11. Control System Security Center. http://www.css-center.or.jp/en/. Cited July 27, 2015.
12. Morris, T., Srivastava, A., Reaves, B., Gao, W., Pavurapu, K., and Reddi, R., A control system testbed to validate critical infrastructure protection concepts, *Int. J. Crit. Infrastruct. Prot.,* 2011, vol. 4, no. 2, pp. 88–103.
13. Davis, C.M., Tate, J.E., Okhravi, H., Grier, C., Overbye, T.J., and Nicol, D., SCADA cyber security testbed development, *38th North American Power Symposium,* 2006, pp. 483–488.
14. Chabukswar, R., Sinopoli, B., Karsai, G., Giani, A., Neema, H., and Davis, A., Simulation of network attacks on SCADA systems, *First Workshop on Secure Control Systems, Cyber Physical Systems Week,* Stockholm, 2010.
15. Kush, N., Clark, A.J., and Foo, E., *Smart Grid Test Bed Design and Implementation,* 2010 (Unpublished).
16. Almalawi, A., Tari, Z., Khalil, I., and Fahad, A., SCADAVT-A framework for SCADA security testbed based on virtualization technology, *IEEE 38th Conference on Local Computer Networks (LCN),* 2013, pp. 639–646.
17. Farooqui, A., Zaidi, S.S.H., Memon, A.Y., Qazi, S., et al., Cyber security backdrop: A SCADA testbed, *Computing, Communications and IT Applications Conference (ComComAp),* 2014, pp. 98–103.
18. Queiroz, C., Mahmood, A., Hu, J., Tari, Z., and Yu, X., Building a SCADA security testbed, *Third International Conference on Network and System Security, Gold Coast, QLD,* 2009, pp. 357–364.
19. Reaves, B. and Morris, T., An open virtual testbed for industrial control system security research, *Int. J. Inf. Secur.,* 2012, vol. 11, no. 4, pp. 215–229.
20. Modbus Application Protocol Specification V1.1a, in *Modbus-IDA,* 2004.
21. Modbus Messaging on TCP/IP Implementation Guide V1.0a, in *Modbus-IDA,* 2004.
22. Ahrenholz, J., Comparison of CORE network emulation platforms, *Military Communications Conference,* San Jose, CA, 2010, pp. 166–171.
23. Pymodbus – A Modbus Protocol Stack in Python. https://code.google.com/p/pymodbus/. Cited July 29, 2015.
24. Ettercap Home Page. http://ettercap.github.io/ettercap/. Cited July 27, 2015.