Integration and Characterization Testing of the LOBO Nuclear CyberSecurity (LOBO NCS) Platform and OpenPLC

Mohamed S. EL-Genk, Timothy Schriener, Asmaa Salem
Institute for Space and Nuclear Power Studies and Nuclear Engineering Department, The University of New Mexico, Albuquerque, NM, USA

Technical Work Scope: *Nuclear Energy–Cybersecurity Research Topics and Metrics Analysis (NE-1).* DOE-NEUP Project 18-15055. DOE Contract No. Nu-18-NM-UNM-050101-01 to University of New Mexico (UNM)

Performance Period: 09-15-2020 to 06-30-2022

Report No. UNM-ISNPS-01-2022
Institute for Space and Nuclear Power Studies, The University of New Mexico, Albuquerque, NM, USA, http://isnps.unm.edu/reports/

July 2022

Executive Summary

The expanding uses of digital instrumentation and control (I&C) systems in commercial nuclear power plants and energy infrastructure raises cybersecurity concerns and emphasizes the need to develop effective tools for investigating potential cyber vulnerabilities. The Nuclear Instrumentation & Control Simulation (NICSim) platform developed at the University of New Mexico's Institute for Space and Nuclear Power Studies (UNM-ISNPS) in collaboration with Sandia National Laboratories (SNL) under a 2018 DOE NEUP award addresses these needs. This platform links physics-based Matlab Simulink models of a representative Pressurized Water Reactor plant and various plant components to emulated or physical Programmable Logic Controllers (PLCs) in the plant's digital I&C systems. As part of this effort, the LOBO Nuclear CyberSecurity (LOBO NCS) platform is developed at UNM-ISNPS to support cybersecurity testing of NICSim in collaboration with Sandia National Laboratories (SNL). The efficient data transfer interface and broker in the LOBO NCS platform manage and coordinate communications between the emulated PLCs and the Simulink models. The LOBO NCS platform could also be used to support academic research and education, and professional training.

The LOBO NCS platform is used to investigate the response of physics-based Simulink models of a representative PWR, and components linked to multiple emulated PLCs during nominal operation transients and when the PLCs are the target of simulated Modbus TCP False Data Injection Attacks (FDIAs). The linked Simulink models of a representative PWR plant and the emulated PLCs in the LOBO NCS platform are used to simulate a reactor startup scenario both without and with a simulated FDIA on the Pressure PLC. The simulated FDIA repeatedly overwrites a false low system pressure to the PLC's holding register. The simulated FDIA initially caused a rapid increase in the system pressure by manipulating the Pressure PLC to increase the power to the submerged proportional heaters and turn on the submerged backup heaters in the pressurizer. The other emulated PLCs linked to the Simulink models attempted to maintain the values of the operation state variables of the plant within their preprogramed setpoints. This limited the impact of the simulated FDIA on the state variables of other plant components.

Investigations of the effects of an FDIA on the emulated PLC show that the input holding register is not successfully overwritten 100% of the time. To understand this behavior this research investigated the effects of the programmed input scan time on the responses of an Allen-Bradley hardware PLC and an emulated PLC using OpenPLC while under simulated FDIAs over a wide range of input scan times. The simulated FDIAs attempted to manipulate the responses of both the Allen-Bradley hardware PLC and the emulated PLC. The first FDIA simulated attempts to overwrite the value of the system pressure to the holding registers of the PLCs to force them to activate and increase the electrical power to the immersed proportional heaters to their peak value and turn on the immersed backup heaters in the pressurizer of a representative PWR plant. The second simulated FDIA attempts to overwrite the holding register of the water spray control function to disable the water spray into the pressurizer during successive surge-in and surge-out transients.

Results show that the input scan time significantly affect the responses of the emulated and hardware PLCs to the simulated Modbus FDIAs. During normal operation, the responses of both PLCs are close, regardless of the value of the input scan time. However, when subject to simulated FDIAs, the percentage of the successful overwrites of the PLCs increases with increased the input scan time. For the same input scan time, the registers of the Allen-Bradley

hardware PLC successfully overwritten at higher percentage of the time compared to the emulated PLC using the OpenPLC runtime. This difference is attributed to the differing means the two PLCs use to manage Modbus TCP communication.

The present results demonstrate the successful integration of the emulated PLCs using OpenPLC within the developed LOBO NCS platform and the capability of this platform for simulating and investigating potential cyber-compromises of emulated PLCs in a representative PWR plant. The modular and versatile LOBO NSC platform can support the development of next generation cybersecurity and autonomous control technologies and methods for terrestrial nuclear reactor power plants, space nuclear power systems, and other power and energy systems. Investigation of the PLCs' responses to simulated FDIAs show that the configuration settings such as the input scan time and the communication routines impact response to a cyberattack scenario. Researchers can use this information to either help configure the PLC to be less susceptible to a potential cyber-compromise or for a particular controller design identify characteristic signs that the PLC targeted by a cyberattack.

List of Contents

List of Figures

List of Tables

Abbreviations

A-B	Allen-Bradley
DCS	Distributed Control System
DOE:	Department of Energy
FDIA	False Data Injection Attack
GUI	Graphical User Interface
IAPWS:	International Association for the Properties of Water and Steam
I&C:	Instrumentation and Control
ICS:	Industrial Control System
I/O:	Input-Output
LOBO NCS	LOBO Nuclear CyberSecurity
ManiPIO	Manipulate Process Input/Output
NEUP:	Nuclear Engineering University Program
NICSim:	Nuclear Instrumentation and Control Simulation
PLC:	Programmable Logic Controller
PWR:	Pressurized Water Reactor
PZ	Pressurizer
RTU	Remote Terminal Unit
SG:	Steam Generator
SNL:	Sandia National Laboratories
TCP:	Transmission Control Protocol
TCP/IP:	Transmission Control Protocol over Internet Protocol
UNM-ISNPS:	University of New Mexico's Institute for Space and Nuclear Power Studies
VLAN	Virtual Local Area Network
VM:	Virtual Machine

1. Introduction

The benefits of introducing digital Instrumentation and Control (I&C) systems in commercial nuclear reactor power plants include enhanced safety, supporting power uprates, and increasing the load factor by reducing the frequency of operation transients with scram. However, the introduction of digital systems such as Programmable Logic Controllers (PLCs) and Distributed Control Systems (DCSs), raises potential cybersecurity vulnerabilities of manipulating sensors' measurements and control signals (Nuclear Energy Institute 2010; National Research Council 1997). Industrial Control Systems (ICSs) traditionally utilize isolated networks from protection from outside cyber-attack. However, recent sophisticated cyber-campaigns on critical industrial, energy, retail and financial infrastructures have shown that isolation alone is insufficient to penetrate even air-gapped secure networks (Karnouskos 2011; Nuclear Energy Institute 2010). A potential cybersecurity event targeting digital PLCs in I&C systems are False Data Injection Attacks (FDIAs). For PLCs with analog inputs, an FDIA attempts to modify the analog current and voltage sensor signals to manipulate the input or output control signals of the PLCs. For a networked PLC receiving digital signals, an FDIA could send false data either by mimicking or taking control of a connected Remote Terminal Unit (RTU) or another PLC in the network. Consequently, the false data manipulates the functions of the targeted PLCs to send control signals contrary to those for nominal plant operation.

In response to such threats researchers are developing capabilities with varying degrees of inclusivity and breadth to investigate cyber vulnerabilities of nuclear power plants. Zhang and Coble (2020) have developed a toolkit, which links physical PLCs to a Pressurized Water Reactor (PWR) simulator as hardware-in-the-loop. This toolkit is used to investigate the effects of a false data injection on the transient response of the feedwater control PLC for the steam generator and to record the digital signatures of the PLC while under cyberattack attack (Zhang and Coble 2020). The Asherah Nuclear Power Plant Simulator developed recently as a multi-national IAEA effort to conduct cybersecurity investigations of nuclear power plants (Busquim E. Silva, et al 2020). This simulator links a transient model of a representative PWR plant to the I&C systems' PLCs either as simulated models or physical hardware integrated as hardware-in-the-loop. Compared to high fidelity emulation models of the PLCs, the low fidelity simplified models are fast running, but would be ineffective in investigating cyber-vulnerabilities of the software and firmware. Conversely, physical PLC hardware-in-the-loop offers the highest fidelity but is costly and difficult to scale up for cybersecurity research. On the other hand, scaling up the emulated I&C systems can be done easily, set up quickly and torn down in secure, sandboxed virtual testing environments.

The University of New Mexico's Institute for Space and Nuclear Power Studies (UNM-ISNPS) in collaboration with Sandia National Laboratories (SNL) developed the LOBO Nuclear CyberSecurity (LOBO NCS) Platform to investigate cyber-vulnerabilities of nuclear reactor plants (El-Genk, et al., 2021; El-Genk and Schriener 2022). This platform links MATLAB Simulink (The Mathworks 2020) physics-based models of a representative PWR plant and various components to either emulated or physical hardware PLCs in the digital I&C systems

(Fig. 1.1). The LOBO NCS platform is modular and extendable to other plant types and to support academic education and research and professional training.

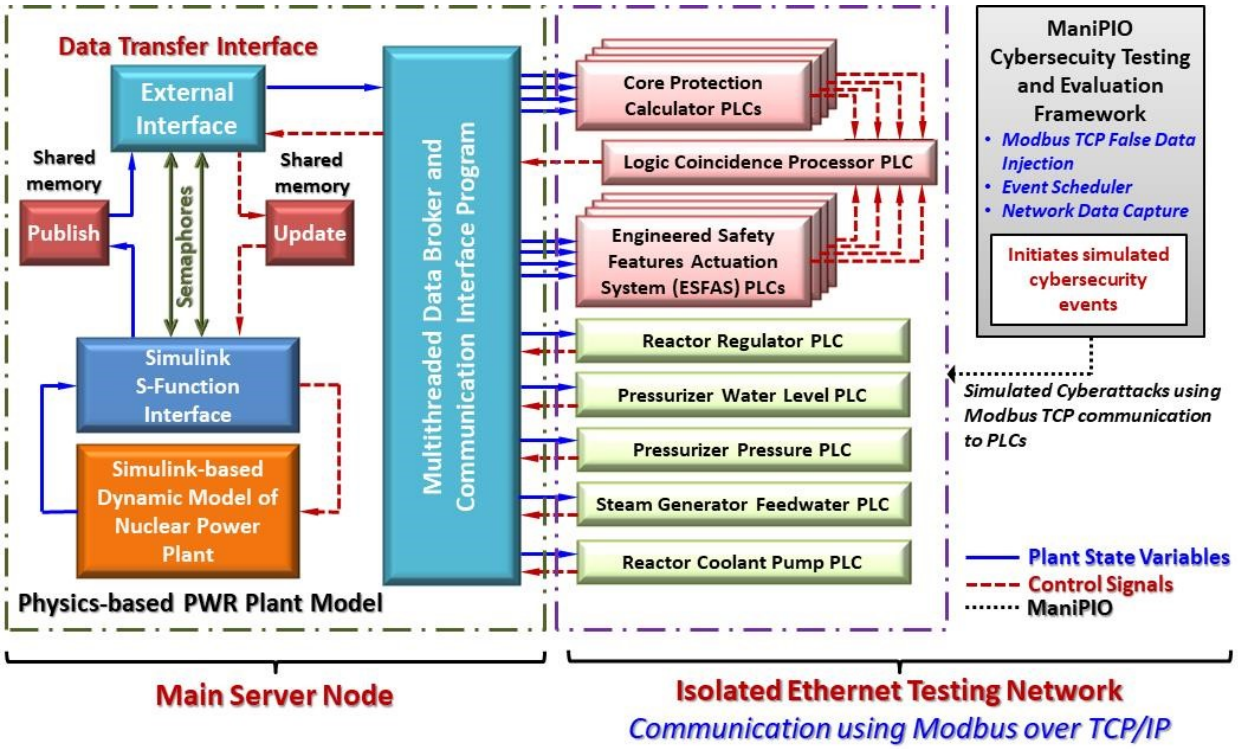

Fig. 1.1. A schematic of the LOBO NCS platform with ManiPIO for simulating FDIAs (El-Genk, et al., 2021).

The Simulink models in the LOBO NCS platform communicate to emulated PLCs in the I&C systems using a fast and reliable data transfer interface and broker program. The user-friendly graphical interface with plotting capabilities provides a real-time display of simulation results. The Manipulate Process Input/Output (ManiPIO) framework in the LOBO NCS platform initiates simulated cyberattacks on the PLCs (Fig. 1.1). This framework incorporates modules for simulating cyberattacks on the PLCs and for capturing and inspecting network traffic for further analysis. The ManiPIO program designed as a research tool using open-source tools and common python libraries. The emulated PLCs each run within separate Virtual Machines (VM) located on multi-processor server nodes connected to the Ethernet network. Physical hardware PLCs connect to the network in place of, or in conjunction with, the emulated PLCs. The Modbus TCP protocol manages the communications between the PLCs and the data on an isolated network.

The LOBO NCS platform integrates the developed emulated PLCs with the physics-based Simulink-Matlab models of a representative PWR plant and components and characterizes their operation when linked to the Data Broker and Communication Interface Program (Fig. 1.1). Evaluating this integration and demonstrating coupling of the developed physics-based models

and the emulated PLCs are important to the development of the platform. The research objectives detailed in this report are to accomplish and demonstrate the integration of the developed emulated PLCs detailed in the submitted milestone report entitled, "*Emulated Programmable Logic Controllers for the Protection and Safety Monitoring and Operation I&C Systems in a Representative PWR Plant for Cybersecurity Applications,"* (El-Genk, et al. 2020a) and the physics-based PWR plant component models detailed in the submitted report entitled "*A Physics-based, Dynamic Model of a Pressurized Water Reactor Plant with Programmable Logic Controllers within the LOBO NCS platform for Cybersecurity Applications,"* (El-Genk, et al. 2020b). The functionality of the integrated PWR plant model and the emulated PLCs demonstrated using the LOBO NCS platform. This simulated an operation transient of a representative reactor startup scenario both during nominal conditions and with the Pressure PLC of the pressurizer subject to a simulated Modbus TCP FDIA.

Section 2 - Transient Modeling of a Representative PWR Plant with Emulated PLCs in the LOBO NCS Platform, evaluates the integration of the developed emulated PLCs with the Matlab Simulink model of a representative PWR plant. The presented research in this section investigates the transient responses of the Simulink model of an integrated PWR plant controlled by the full set of developed emulated PLCs (Fig. 1.1) during a nominal operation transient and with one of the PLCs subject to a simulated Modbus TCP FDIA using the ManiPIO framework. The results of this research demonstrate the capability of the LOBO NCS platform and of linking a physics-based dynamic model of a representative PWR plant to a multitude of emulated PLCs controlling different semi-autonomous functions. The presented results are for a simulated nominal startup transient for a representative PWR and for the same startup scenario but with a simulated FDIA targeting pressure PLC to manipulate it to increase the system pressure in the primary loop.

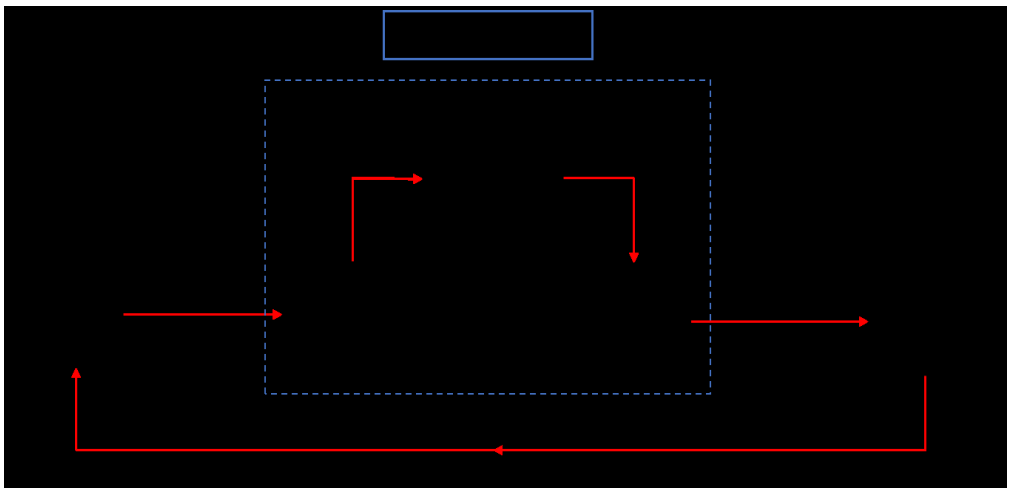

Fig. 1.2. A block diagram of the basic functions performed during a PLC's scan time cycle.

In addition to demonstrating integrated operation of the developed components of the LOBO-NCS platform, this report presents the results investigating the response of the emulated PLCs when subjected to Modbus TCP FDIAs generated by the ManiPIO framework. Previous

investigations of Modbus TCP FDIAs using the LOBO NCS Platform showed that simulated FDIA overwriting the Modbus holding registers on the PLC was not able to overwrite the value 100% of the time (Schriener and El-Genk 2021). This behavior is observed with both the emulated PLC and the commercial Allen-Bradley PLC coupled to the LOBO NCS platform when targeted by simulated FDIAs using the ManiPIO program.

PLCs perform their control operations in a repetitive cycle known as the scan cycle. During each cycle, the PLC performs three specific operations (Figure 1.2), namely: (a) reading (or scanning) the input values, (b) executing the control logic program using the input values, and (c) writing the output values for subsequent control response. The LOBO NCS data broker and communication interface program send calculated plant state variables to the input holding registers of the PLCs using Modbus TCP communication at regular intervals tied to the simulation timestep of the simulation. During simulated FDIAs the ManiPIO program sends write requests to the targeted holding register(s) on the PLC at a frequency independent to that of the LOBO NCS data broker.

The interplay between these different, unsynchronized periodic processes could be contributing to the observed failure of the simulated FDIA to consistently overwrite the PLCs' holding registers. Research results investigating the effect of the input scan time of an emulated and a hardware PLC on the response while under simulated cyberattacks is detailed in ***Section 3 - Characterization of Simulated False Data Injection Attacks (FDIA) on Emulated and Hardware Programmable Logic Controllers***. This effort investigates and compares the responses of an emulated PLC and a commercial Allen-Bradley PLC while configured for the pressurizer in a representative PWR plant during nominal steady state and transient operations and while the PLCs are under a series of simulated Modbus TCP FDIAs initiated by the ManiPIO framework. The conducted parametric analyses quantify the effects of changing the input scan time and the register targeted on the percentage of successful overwrite attempts during a simulated FDIA. It also compares the responses of the PLCs control actions and the resulting operation of the physics-based pressurizer model.

Lastly, the work in ***Appendix A – Characterization of OpenPLC in LOBO NCS*** investigates the performance of the open-source OpenPLC software for the emulated PLCs. It determines the scan time of the emulated PLCs using OpenPLC and examines the consistency between the actual scan time of the control program and the specified input scan time. While commercial hardware PLCs typically keep highly consistent scan times using their integrated real time clocks, it is necessary to understand how closely the emulated PLCs replicate this operation. Based on the results options are investigated for improving the consistency of the actual scan time of the OpenPLC software by modifying the source code. Results also quantify the communication timing between the OpenPLC and an external communication interface on the LOBO NCS platform. Investigated are the effects of the simulation timestep, the PLC scan time, and the communication network on the timing and reliability of the Modbus TCP communication between the PLC and a Matlab Simulink model in the LOBO NCS testing network.

2. Transient Testing of Developed Representative PWR Plant with Emulated PLCs in the LOBO NCS Platform

The LOBO NCS platform links a multitude of independent PLCs to Matlab Simulink, physics-based models of a representative nuclear power systems. The PLCs successfully evaluated independently while linked to physics-based components models and the integrated plant model (El-Genk, et al. 2020). This section demonstrates integration of the developed LOBO NCS platform to the developed PLCs in the I&C system controlling the functions of a representative PWR plant model. Results compare the responses of the plant and the PLCs during nominal operation and when the emulated PLCs are targets to simulated Modbus TCP FDIAs initiating by the ManiPIO program (El-Genk, et al. 2021). The simulated FDIAs target the input Modbus holding registers of one of the emulated PLCs in the I&C system to affect operation during a simulated operational transient of a reactor startup. Results evaluate the LOBO NCS platform's ability to simultaneously communicate the plant state variables to and receive command signals from multiple emulated PLCs. This is while ensuring smooth stable operation of the connected Simulink PWR plant model during the transient both with and without an FDIA.

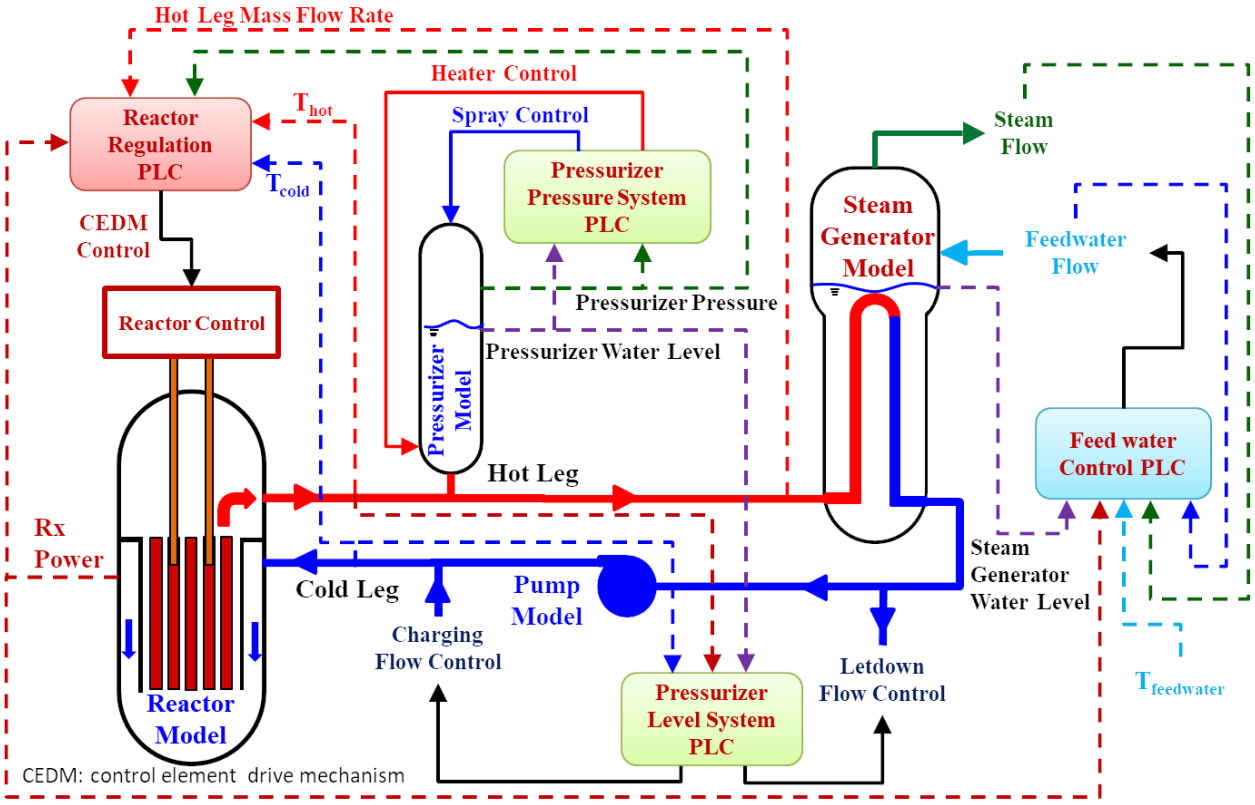

Fig 2.1. Block diagrams of integrated physics-based models of a representative PWR plant and various components with the emulated PLCs in the I&C systems (El-Genk and Schriener 2022).

2.1. Emulated PLCs in I&C System

Figure 2.1 shows the LOBO NCS platform configured with the emulated PLCs of the plant Operation I&C system. The emulated PLCs are the Pressure PLC, the pressurizer Water Level PLC, the steam generator Feedwater PLC, the Reactor Regulation PLC, and the Pump PLC (Fig. 1.1). These PLCs receive the calculated values state variables by the Simulink models of the plant's components and send back control feedback signals to the integrated plant model. This section summaries the functions of these PLCs, detailed elsewhere (El-Genk, et al. 2020). The Reactor Regulation PLC monitors and adjusts the reactor thermal power within specified setpoints by the operators. It compares the differences between the desired reactor power and that calculated by the coupled reactor point-kinetics and primary loops thermal-hydraulics models.

The Pressure PLC monitors and maintains the system pressure in the primary loop within preprogramed setpoints. It controls the electrical power level to the immersed proportional heaters, turns on or off the electrical power to the immersed backup heaters, and changes the opening of the water spray nozzle. This PLC also intermittently opens the relief valve in the pressurizer when system pressure surpasses a maximum setpoint. The water level PLC monitors and regulates the water level in the pressurizer and the total water inventory in the primary loops. The water inventory in the primary loops adjusts by controlling the rates of water inflow into the primary loop from the charging pumps and water outflow through the letdown valves. The pressurizer's water level and pressure PLCs work together to adjust the pressure and water inventory in the primary loops.

The steam generator feedwater PLCs control the water inventory in the Steam Generators (SGs) of the plant to ensure that water covers the U-tube bundles in the SGs. These PLCs monitor the measured water level in the downcomer of the SGs and adjust the rate of feedwater flow. They also accommodate the change in the steam generation rate in response to a change in the electrical load demand. Adjusting the opening of the throttle valve between the feedwater pumps and the water injection into the steam generators control the feedwater rate. The pump PLC regulates the shaft rotation speed of the reactor pumps connected to the cold legs of the primary loop. This PLC uses the calculated water flow rate and total pressure losses in the primary loops to determine the target shaft rotational speed and adjusts those of the pumps to match the target value.

2.2. Simulated Nominal Startup Transient

This subsection presents the results of a simulated reactor startup using the integrated Simulink model of a representative PWR plant linked to emulated PLCs in the Operation I&C system (Figs. 1.1. 2.1). Emulated PLC employs an open-source architecture using the OpenPLC runtime software (Alves and Morris 2018) within a Raspian OS Virtual Machine (VM). In addition to the PLCs control inputs, the model follows a user input script of the withdrawal of the control assemblies in the reactor core and the soluble boron concentration in the coolant. The Simulink model uses a fixed discrete simulation timestep of 20 ms and the emulated PLCs

configured with an input scan time of 50 ms. The LOBO NCS data transfer interface (Fig. 1.1) controls and synchronize the progression of the Simulink simulation with a real-time (or wall time).

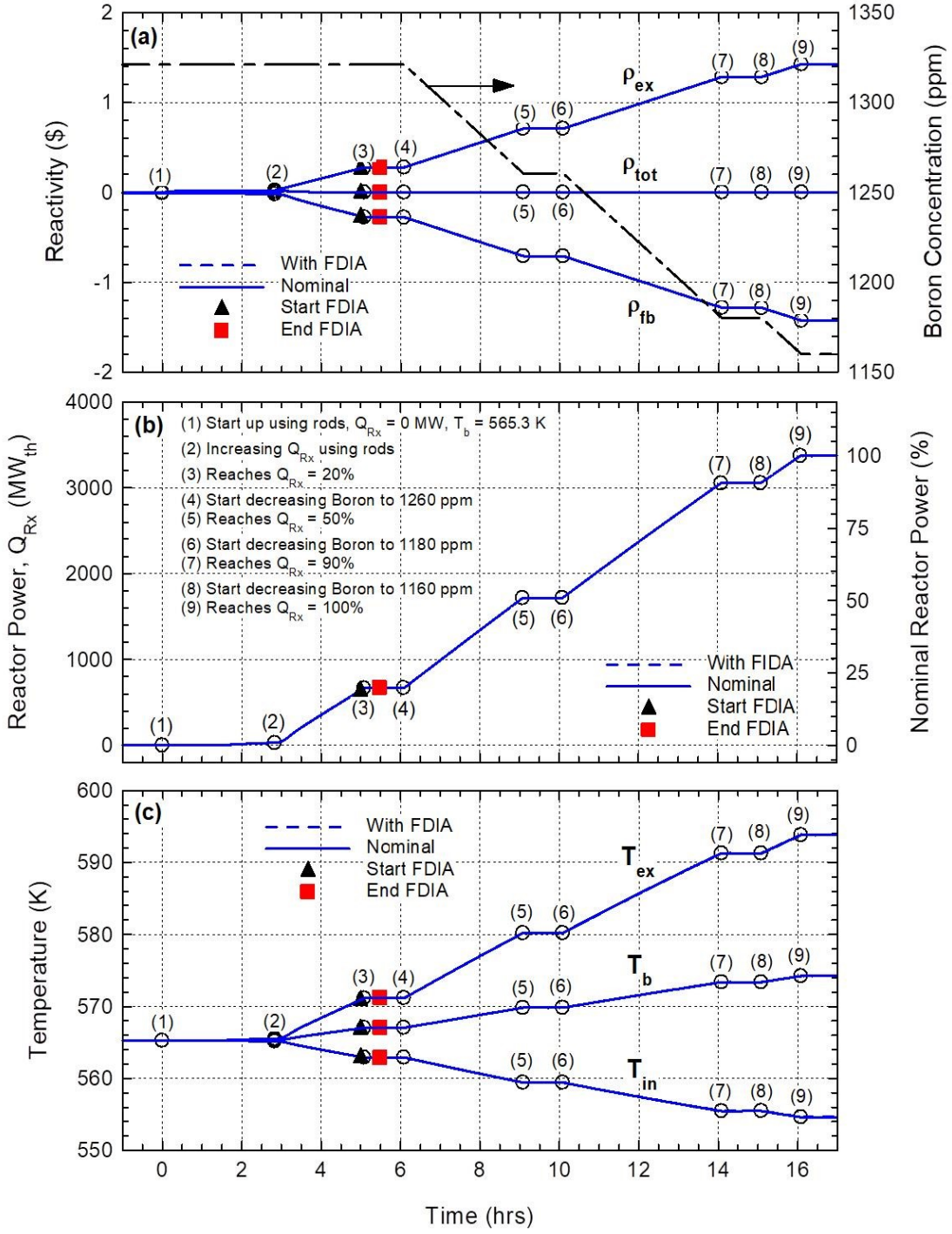

Fig. 2.2. Calculated state variables of a representative PWR plant using the reactor Simulink model during simulated startups without and with a FDIA (El-Genk and Schriener 2022).

Figures 2.2 - 2.6 present the results of the simulated startup transient. The plant start transient begins at a hot critical condition with: (a) all four reactor coolant pumps are running at shaft rotation speed of 1,053.1 rpm and dissipating 15 MW_{th} into the circulating water in the primary loops, (b) the pressurizer operation controlled by the Pressure and Water Level PLCs to maintains a system pressure of 15.686 MPa, (c) the circulating water in the primary loops is a mean temperature of 565 K due to the thermal dissipation from the reactor pumps, and (d) the heat from the hot legs of the primary loops is removed in the steam generators to the circulating water from the secondary loop. The simulated reactor startup begins by withdrawing the control element assemblies in the core to insert 2.07¢ of external reactivity (point 1 in Figs. 2.2-2.6) and increase the reactor power to 2% of its nominal full power value (point 2 in Figs. 2.2-2.6). The rate of steam supply by the SG increases commensurate with the increase in the reactor thermal power (Fig. 2.5a).

At point (2), the control element assemblies in the reactor core withdrawn further to insert a total of 27.77¢ of external reactivity and raise the reactor thermal power to 20% of nominal (point 3 in Figs. 2.2-2.6) and increase the rate of steam generation to the turbine on the secondary side of the plant. These conditions are held steady for one hour to allow the system to achieve steady state operation at this power level. Subsequently, the position of the control element assemblies in the core holds constant with an additional external reactivity insertion by diluting the concentration of the soluble boron in the circulating primary coolant in the reactor core.

The boron concentration initially at 1,321 ppm decreases to 1,260.6 ppm to bring the reactor power up to 50% of nominal (point 5 in Figs. 2.2-2.6). This condition hold constant for one hour for the operators to recalibrate the nuclear instrumentation. Subsequently, the soluble boron concentration in the primary loop decreases further to 1,180.1 ppm to increase the reactor power to 90% of nominal (point 7 in Figs. 2.2-2.6). This power level hold constant for one hour to recalibrate the nuclear instrumentation prior to bringing the reactor to full power. To do that, the boron concentration in the primary loop decrease further to 1,060 ppm and the reactor thermal power eventually reaches 100% of nominal or 3,373 MW_{th} (point 9 in Figs. 2.2-2.6).

Figures 2.2- 2.6 show the calculated transient values of select state variables of the plant during the described startup sequence. Results in Fig. 2.2a-c are of the state variables calculated by the developed reactor Simulink model in the LOBO NCS. Results in Fig. 2.3a-c are of the pressurizer model linked to the Pressure PLC. The results in Fig. 2.4a-c are of the primary loop model linked to the pressurizer's Water Level PLC.

Figures 2.5a-b shows the calculated state variables by the SG model linked to the Feedwater PLC and Fig. 2.6 is of the calculated water flow rate and the shaft rotation speed of the primary pump model linked to the pumps PLCs. During the simulated start up transient the increase in the reactor thermal power (Fig. 2.2b) increased the flow rate and exit temperature of the coolant circulating through the reactor core (Fig. 2.2c). The increased temperature introduces a negative reactivity feedback, which equals the positive external reactivity insertion, and maintains the reactor core critical with near zero total reactivity (Fig. 2.2a). The corresponding decrease in the temperature of the coolant entering the reactor core is due to the increased rate of steam

production in the SG to the turbine in the secondary side of the PWR plant (Fig. 2.5a). As shown in Fig. 2.2c, the increase in the exit temperature of core coolant is more than the decrease in its inlet temperature, increasing the bulk temperature of the coolant/moderator in the core. Such increase in temperature introduces a negative temperature reactivity feedback (Fig. 2.2a).

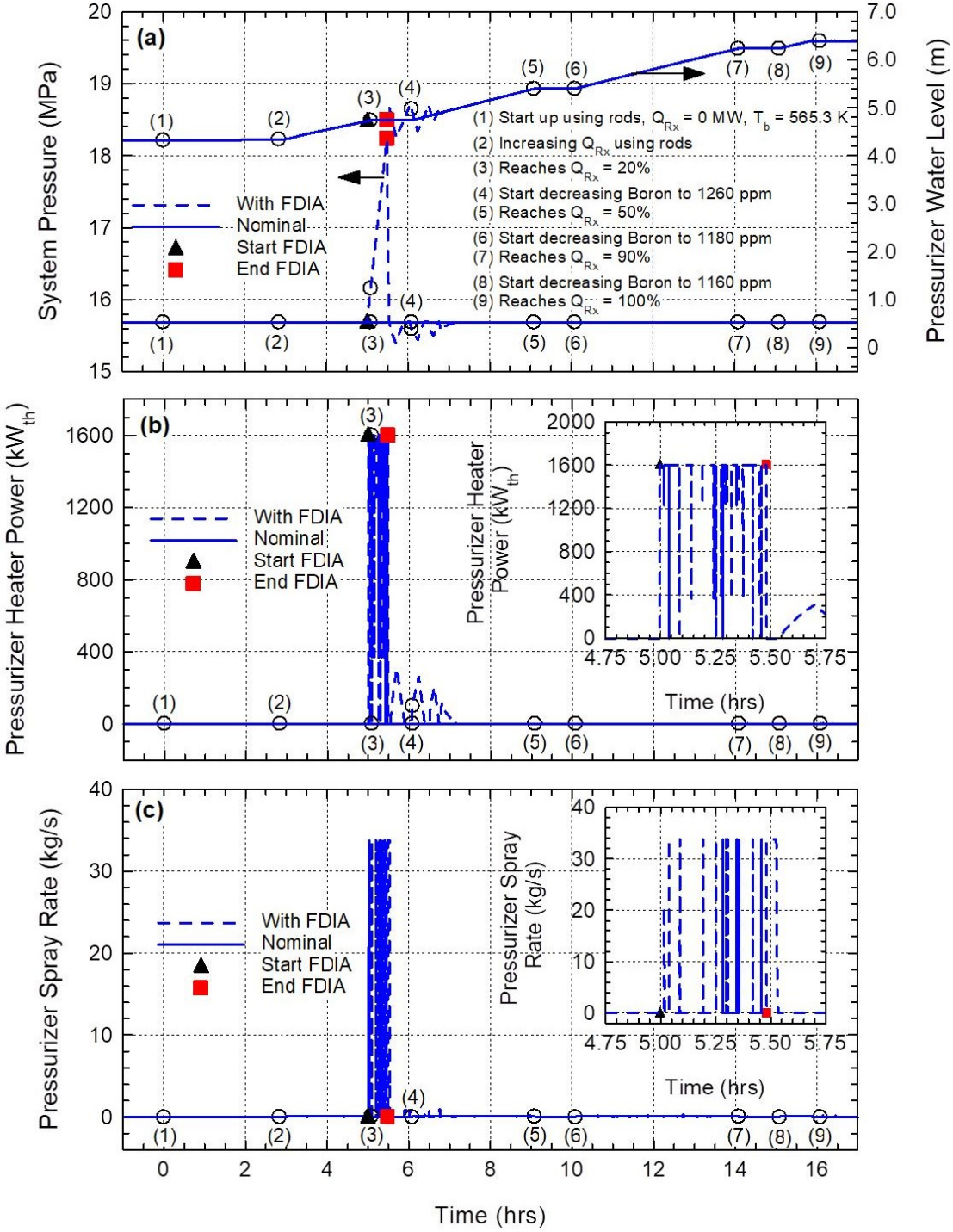

Fig. 2.3. Calculated state variables of a representative PWR plant using the pressurizer Simulink model linked to the Pressure PLC during a simulated startup without and with an FDIA (El-Genk and Schriener 2022).

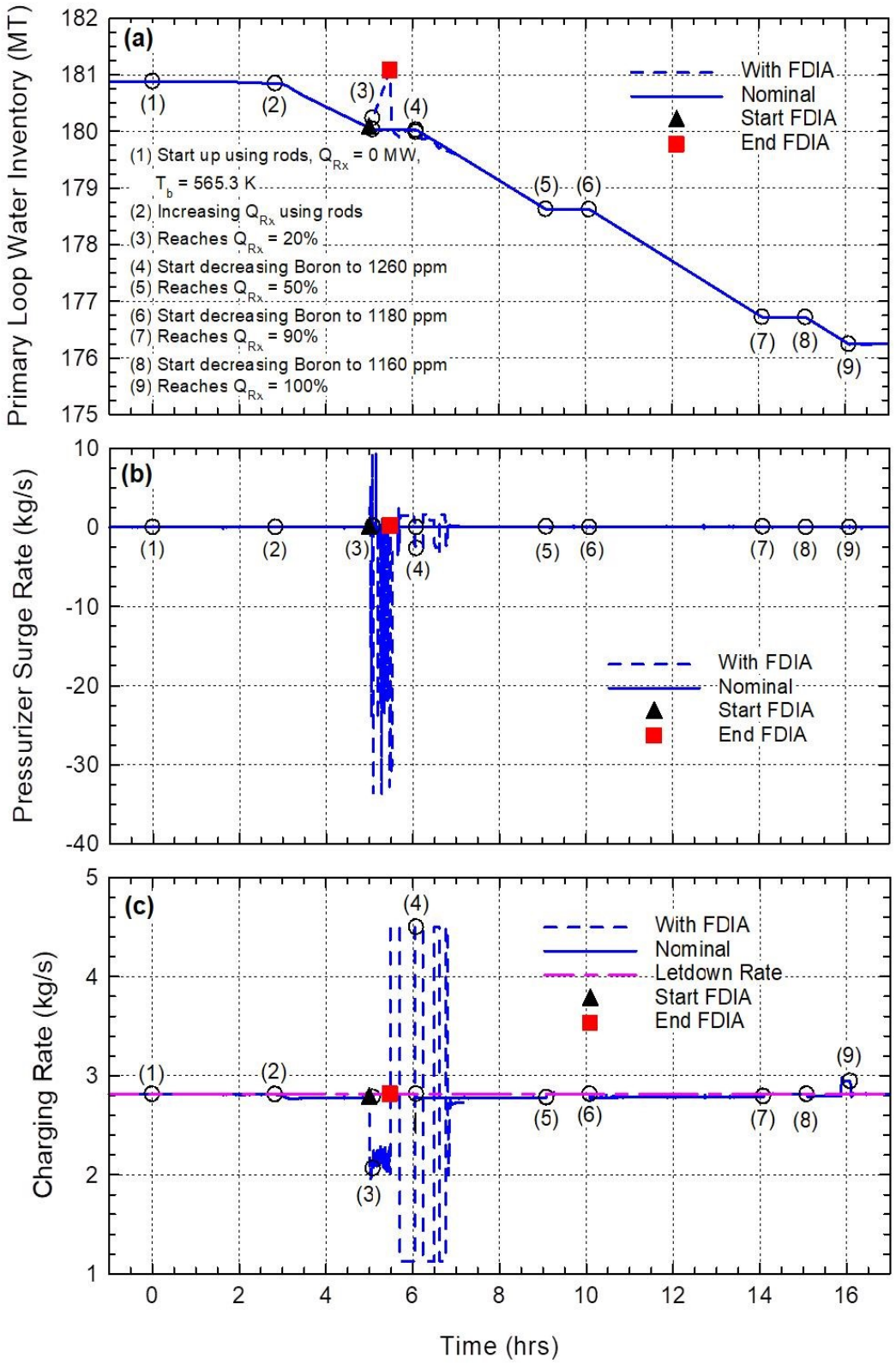

Fig. 2.4. Calculated state variables of a representative PWR plant using the primary loop Simulink model linked to the pressurizer Water Level PLC during a simulated startup without and with an FDIA (El-Genk and Schriener 2022).

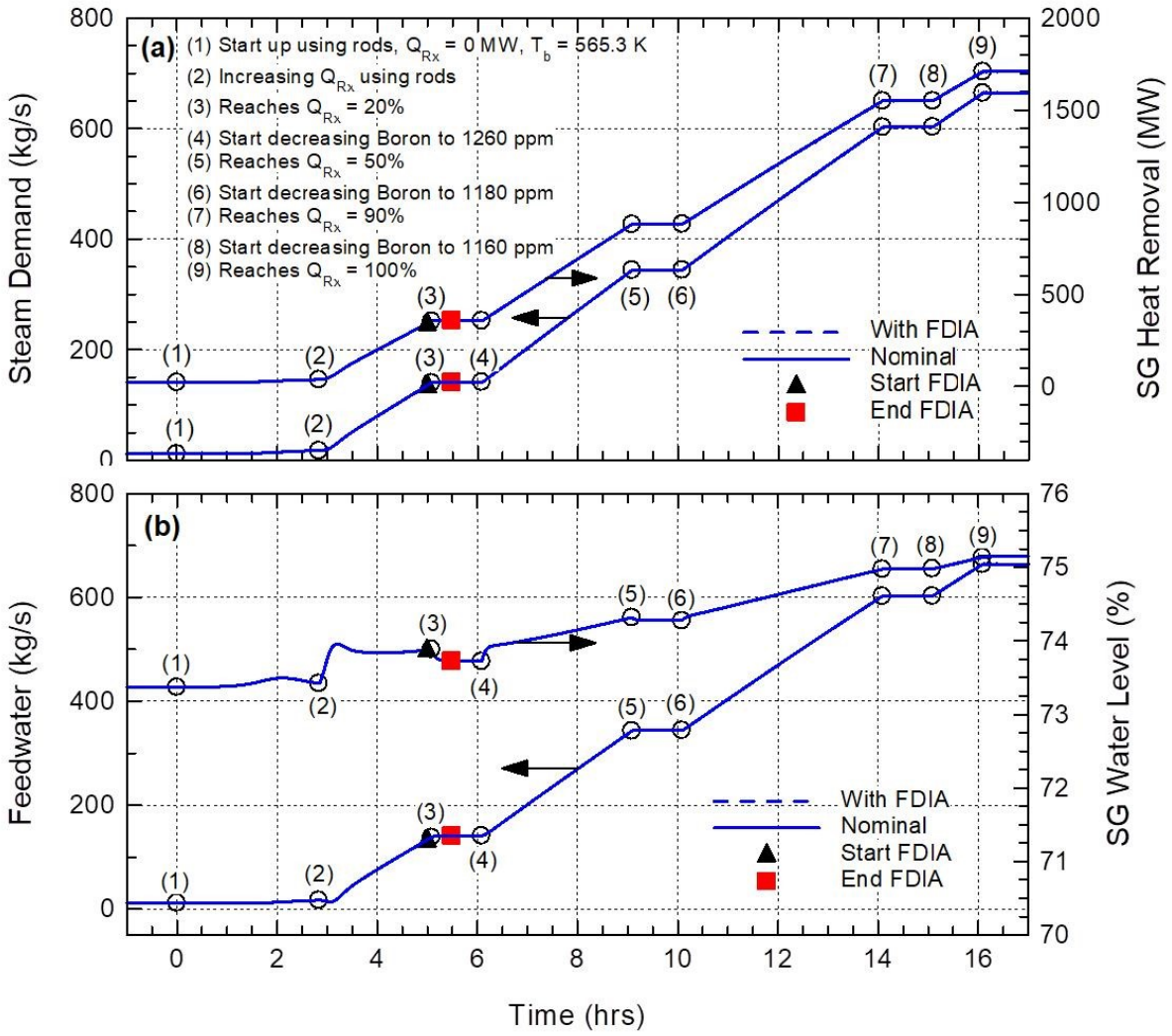

Fig. 2.5. Calculated state variables of a representative PWR plant using the SG Simulink model linked to Feedwater PLC during a simulated startup without and with an FDIA (El-Genk and Schriener 2022).

The Feedwater PLC adjusts the rate of water injection to maintain the water level in the SG at the preprogramed setpoint (Fig. 2.5b). The magnitude of the adjustment depends on the reactor thermal power during the simulated startup transient. During the startup sequence the Pressurizer PLC maintains the system pressure within preprogramed setpoints. As a result, the system pressure changes slightly relative to the nominal value of 15.686 MPa (Fig. 2.3a). This is due to small adjustments of the electrical power to the immersed proportional heaters (Fig. 2.3b) and of the water droplets spray in the pressurizer (Fig. 2.3c) (El-Genk, Altamimi, and Schriener 2021).

The thermal expansion or water in the primary loop during the simulated startup scenario causes a surge-in of water into the pressurizer (Fig. 2.4b). In response, the Water Level PLC for the pressurizer decreases the charging rate of the primary loop to maintain the water level in the pressurizer within preprogramed setpoints (Figs. 2.3a and 2.4c). Decreasing the charging rate below the letdown rate in the primary loops (Fig. 2.4b) decreases in the total coolant inventory in

the primary loops (Fig. 2.4a). The Water Level PLC increases the setpoint commensurate with the reactor bulk coolant temperature (Fig. 2.2c). This helps accommodate the thermal expansion of the primary loops coolant while minimizing the adjustment of the charging rate (Figs. 2.2a and 2.4c). Over time, the water level in the pressurizer follows similar profile to that of the reactor thermal power, increasing linearly during the simulated external reactivity insertion periods (Figs. 2.2a-b, 2.3a).

Rising water level in the pressurizer increases the system pressure (Fig. 2.4a) prompting the Pressure PLC to decrease the power to the submerged electrical heaters and actuate the droplets spray of subcooled water from the cold leg into the saturated steam region of the pressurizer (Figs. 2.4b-c). The small increases in the charging rate of the primary loops between points (8) and (9) during the startup scenario is because the PI controller of the Water Level PLC over adjusts the charging rate when the reactor thermal power increases from 90% to 100% full power (Fig. 2.5c). To compensate for these differences, the PLC adjusts the charging rate to align the water level with the preprogramed setpoint (Fig. 2.4a).

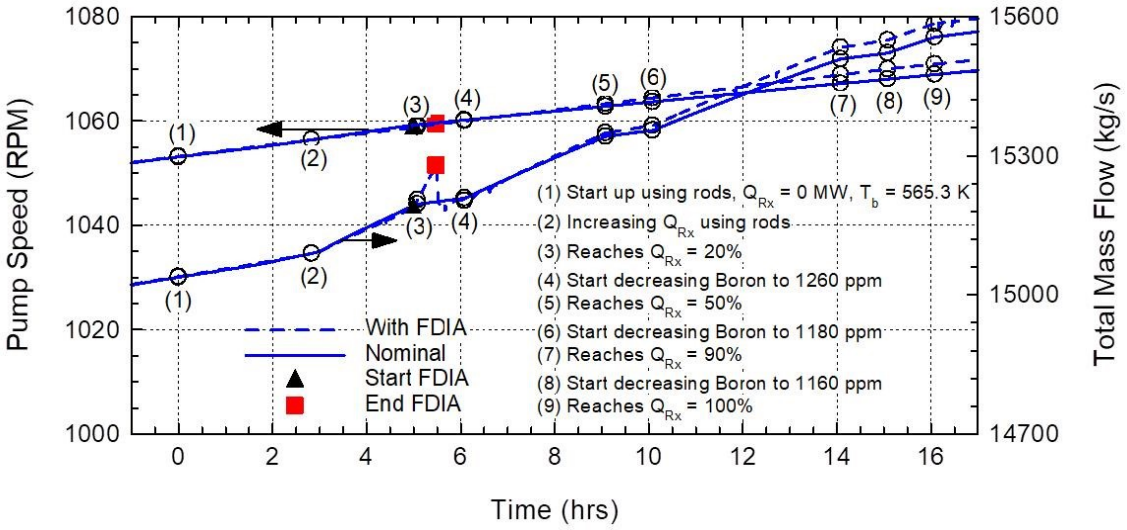

Fig. 2.6. Calculated state variables of a representative PWR plant using the linked Simulink model of the reactor coolant pump to the pumps PLCs during a simulated startup without and with an FDIA (El-Genk and Schriener 2022).

The pump PLC increases the shaft rotation speed during the simulated reactor startup to increase the mass flow rate of the water coolant through the core commensurate with the increase in the reactor power (Fig. 2.7). The increased shaft rotation speed increases the pump head and hence the flow rate of the water coolant in the primary loops (Fig. 2.7). The pump shaft speed is 1,053 rpm at the start of the simulated startup (point 1 in Fig. 2.7) increases to 1,069 rpm at the end of the external reactivity insertion at point (9). It continues to increase slightly at the end of the 17-hr long startup transient and levels off later than other controlled parameters. At the end of the simulated reactor startup transient the integrated PWR plant Simulink model the plant operates at nominal full power conditions.

2.3. Simulated Startup with an FDIA

The next subsection investigates the transient response of the integrated PWR plant Simulink model linked to the PLCs in the I&C system during the same startup scenario described above in Section 2.2, but with the pressurizer's pressure PLC targeted by a simulated FDIA. The simulated cyberattack begins at t = 5 hrs. into the simulated startup transient (solid triangular symbol in Figs. 2.2–2.6) at the end of the second external reactivity insertion. At such point, the control element assemblies are withdrawn from the reactor core to bring the thermal power to 20% of the nominal (Figs. 2.2 and 2.3). The ManiPIO program writes a false low system pressure of 15 MPa to the Modbus input holding register of the Pressure PLC. In response the PLC's logic programming sends commands to turn the electrical powers to the immersed proportional and backup heaters fully on and keep the water spray fully off. The simulated FDIA lasts for 0.5 hr. (solid square symbol in Figs. 2.2–2.6), and beyond which the Pressure PLC returns the plant to its normal operating conditions.

Figures 2.2–2.6 compare the results of the simulated startup of a representative PWR plant with a FDIA targeting the Modbus input holding register of the pressure PLC to those obtained earlier for a nominal startup without a FDIA. Prior to the introduction of the simulated FDIA, the Simulink model of the integrated PWR plant calculates the same state variables as during the nominal startup. After introducing the FDIA, the ManiPIO program continues to repeatedly overwrite the system pressure in the Modbus holding register with an artificially low value of 15 MPa. In response, the Pressure PLC sends a control signal to turn on the proportional heaters to full power and switch the backup heaters on (Fig. 2.3b). Together these heaters supply 1.6 MW_{th} to the water within the pressurizer which increase the rate of flash evaporation into the upper vapor region of the pressurizer to increase the system pressure (Fig. 2.3a).

The simulated FDIA on the Pressure PLC rapidly increases the system pressure for the duration of the FDIA (Fig. 2.3a). This pressure peaks at 18.238 MPa, which is 2.553 MPa higher the nominal pressure setpoint of 15.686 MPa. Despite the high pressure the FDIA manipulates the PLC to keep the heaters on and keep closed the water spray nozzle and the pressure relief valve. During the simulated FDIA the ManiPIO program competes with the data communication program in the LOBO NCS to write to the holding register of the Pressure PLC. At certain times during the simulated FDIA the data communication program write the true pressure value to the PLC's input holding register. These instances correspond to the spikes in the inserts in Figs. 2.3b and c. When they occur, the Pressure PLC attempts to correct the high-pressure by momentarily shutting off the heaters and injecting water spray droplets into the pressurizer to induce condensation and decrease the system pressure (Figs. 2.3a-c). Once the simulated cyberattack again overwrites the false low pressure the water spray shuts off and the submerged heaters turn back on. The brief failures to overwrite the pressure register do not appear to significantly affect the induced rise in the system pressure during the FDIA. The inconsistent overwriting of the PLC's holding register by the Modbus TCP FDIA is the subject of the work presented in Section 3.

The increase in system pressure causes out-surges of the water from the pressurizer into the hot leg of the primary loop (Fig. 2.4b). Conversely, the decrease in the system pressure following the actuation of the water droplets spray causes a surge-in from the hot leg into the pressurizer. During the series of the surge-in and surge-out events the pressurizer's Water Level PLC attempts to change the charging rate of water into the primary loops to keep the water level in the pressurizer at the programmed setpoint (Fig. 2.4c). This results in oscillations in the total water inventory in the primary loops as the Water Level PLC attempts to increase and decrease the charging rate water into the primary loops in response to the decrease and increase in system pressure, respectively (Figs. 2.4a-c). The combined effect of the increased pressure and the adjustments of the water charging rate slightly increases the water inventory in the primary loop compared to that at nominal conditions (Fig. 2.4a).

The changes in the system pressure also affect the water properties in the primary loop, such as the density and viscosity, and hence the mass flow rate of the water coolant by the reactor pumps (Fig. 2.6). The Pump PLC slightly adjusts the shaft rotation speed of the pumps to maintain the flow rates through the reactor core during the startup scenario with a FDIA within preprogramed setpoints. The effects of the simulated FDIA on the reactor coolant inlet and exit temperatures and the state variables calculated by the SG Simulink model in the LOBO NCS are negligible (Figs. 2.2c and 2.5).

Once the simulated FDIA ceases, the Pressure PLC returns the system pressure to its normal value. The PLC acts to rapidly decrease the system pressure by fully opening the water spray nozzle (Figs. 2.3a and c). The Water Level PLC also acts to bring the water inventory in the primary loops back to nominal, in line with the system pressure (Fig. 2.4a and c). The control actions of this PLC dampen the oscillations in the water level until reaching the nominal value (Figs. 2.3a, 2.4c). Most other state variables return to their calculated values during the nominal startup without an FDIA (Figs. 2.2-2.5). The exception is the pumps' shaft rotation speed. This is because the Pump PLC does not adjust downward the shaft speed, which remains slightly higher than nominal during the simulated startup transient with an FDIA (Fig. 2.6).

The present results show that the simulated FDIA on the emulated Pressure PLC significantly increases the system pressure in the primary loop. The FDIA attempted to overwrite the actual system pressure to the input holding register of the pressurizer PLC every scan cycle. Occasionally during the simulated FDIA the nominal pressure value did get through to the PLC and it reacts to try to bring the pressure to within its programed setpoints. In these instances, the effects on the pressure rise during the FDIA are negligible. After the simulated FDIA ends the Pressure PLC restores the pressure by increasing the rate of the water droplets spray into the vapor region of the pressurizer. Simultaneously, the Water Level PLC adjusts the charging rate into the primary loop to bring the water level in the pressure to its nominal value shortly afterwards. During the simulated startup with a FDIA, the responses of the other emulated PLCs in the I&C systems of the PWR plant kept other state variables close to their nominal values during the simulated startup without a FDIA (Figs. 2.2-2.6).

2.4. Summary

This section demonstrates the functionality of the integrated elements of the LOBO NCS Platform during nominal operation and while PLCs in the I&C systems of a representative PWR are subjected to simulated cyberattacks. Investigated is the response of the physics-based Simulink models of a representative PWR, and components linked to emulated PLCs. The ManiPIO software program generated the simulated FDIA targeting the one of the PLC. The simulated reactor startup scenario from a hot zero power critical condition continues until reaching nominal full operation at a reactor thermal power of 3,373 MW_{th}. Emulated PLCs running within separate VMs perform the plants' semi-autonomous control functions. The coupled PLCs to the Simulink PWR plant model use the LOBO NCS data broker and communication interface programs. During the simulated startup, emulated PLCs control the plant's functions to steady state nominal full power at the end of the startup scenario.

Compared are the investigation results of introducing a simulated FDIA targeting the Pressure PLC on the plant operation during the same startup sequence without a FDIA. The ManiPIO program produces a simulated Modbus TCP FDIA which repeatedly overwrote a false low system pressure to the corresponding PLC input holding register. The simulated FDIA initially caused a rapid increase in the system pressure by manipulating the Pressure PLC to increase the power supplied to the submerged proportional heaters and turn on the submerged backup heaters in the pressurizer. The simulated FDIA attempts in every simulation timestep to overwrite the system pressure. However, during the FDIA the communication interface in the LOBO NCS occasionally writes the true value of the system pressure to the PLC. Upon receiving the manipulated low-pressure value, the PLC sends commands to activate the submerged heaters. Similarly, the PLC turns on the water spray into the vapor region of the pressurizer when receiving the true system pressure value. These isolated events did not impact the steady rise in the system pressure during the application of the FDIA on the pressurizer pressure PLC. The next section investigates the behavior of the PLC targeted by a Modbus TCP FDIA and characterizes how the PLC's input scan cycle affects its response to the FDIA.

3. Characterization of Simulated False Data Injection Attacks (FDIA) on Emulated and Hardware Programmable Logic Controllers

Results presented in the previous section show that an FDIA overwriting the Modbus holding registers of the pressurizer Pressure PLC failed to overwrite the system pressure value 100% of the time. This response is like that reported by El-Genk, et al. (2021) for simulated Modbus TCP FDIAs on an emulated PLC and a commercial hardware PLC serving as the pressure PLC. This simulated FDIA was during modeled PWR pressurizer sequential surge-in and surge-out transients. The simulations were for the PLCs running with a 2 ms input scan time (Schriener and El-Genk 2021). The behavior observed with the emulated PLC is not an artifact of the PLC emulation platform but also present for an unrelated commercial-grade and the commercial Allen-Bradley (A-B) hardware PLC. Alves and Morris (2018) observed similar behavior when experimentally investigated the effect of a Modbus TCP FDIA on the operation and responses of a soft PLC developed using the OpenPLC runtime, and commercial hardware PLCs by Siemens, Allen-Bradley, Schneider-Electric, and Omron. The soft and hardware PLCs in their tests are programed with the same controller logic to monitor and count the pulses of repeating sawtooth input signals. The simulated FDIA attempted to overwrite the Modbus register which held the internal counter of the signal pulses. All five PLCs responded differently to the Modbus TCP FDIA. The FDIA on the Omron PLC consistently overwrote the value for the counter. The OpenPLC runtime and the Schneider Electric PLC showed similar behavior to each other with the counter repeatedly reverting to between the true and false values as these PLCs and the simulated FDIA cyberattack competed to writing to the Modbus register. The Allen Bradley PLC alternated between three different values as the counter resets, while the Siemens PLC unaffected by the simulated FDIA used in this study. That the inconsistent overwriting behavior is observed by other researchers shows that it is not unique to the LOBO NCS platform but caused by to the interactions between the PLC, communication program, and the simulated FDIA.

Factors that affect the observed response of the PLCs when targeted with simulated FDIAs include: (a) how the PLC program uses the memory register for storing input or output values or as an internal storage or counter; and (b) the configuration settings of the PLC, such as its programmed input scan time. Therefore, it is desirable to understand how these, and other factors could affect a PLC's response to a cyberattack and assess potential impact on the operation, safety and security of a nuclear plant and various industrial and energy infrastructures. The work presented in this section aims to: (i) investigate and compare the responses of both an emulated PLC using OpenPLC and a commercial-grade hardware Allen-Bradley PLC, configured as pressure PLC for the pressurizer in a representative PWR plant during nominal steady state and transient operations and while the PLCs are under a series of simulated Modbus TCP FDIAs; and (ii) perform parametric analysis to quantify the effects of changing the input scan time and the register targeted by a simulated FDIA on the responses of the PLCs and the operation of the pressurizer, and (iii) investigate the effect of the FDIA on the OpenPLC with source code modified to improve consistency with the input scan time (Section A.1.2).

The LOBO NCS platform linked a Simulink physics-based transient model of the pressurizer

(El-Genk, Altamimi, and Schriener 2021) to an emulated PLC based on the open-source OpenPLC runtime and to an Allen-Bradley Micrologix commercial hardware PLC (Schriener and El-Genk 2021). The pressurizer model simulates an operational transient involving sequential surge-in and surge-out events of water from and to the hot leg of the primary coolant loop of a representative PWR plant. The pressure PLC controls the rate of water spray into the pressurizer and the electrical power to the submerged proportional and backup heaters in the water region at bottom of the pressurizer. The calculated responses of the emulated OpenPLC and the Allen-Bradley hardware PLC compared for different values of the input scan time, during nominal operation and when their input and output holding registers are the target of two simulated FDIA scenarios. In one simulated FDIA scenario the ManiPIO program sends a false low-pressure value to the input Modbus holding registers of the pressure PLC. In the second simulated FDIA the ManiPIO program alters the output holding register of the control signal to the PLC to prevent it from activating the water spray into the pressurizer. The next section briefly describes the testing setup used in these investigations.

3.1 LOBO NCS Platform Setup

The LOBO NCS platform detailed in previous reports and publications (El-Genk and Schriener 2022; El-Genk et al. 2021), links dynamic Simulink models of a representative PWR plant and various components to emulated physical PLCs in the digital I&C system in an isolated Ethernet testing network (Fig. 3.1). The integrated PWR plant Simulink model links physics-based models of the primary loop and various plant components and calculate state variables during nominal and simulated operation transients. An efficient synchronous data transfer interface communicates calculated values o and from an external interface program (Hahn, Schriener, and El-Genk 2020). A POSIX shared memory inter-process communication transfers the generated data by the Simulink physics models through a Simulink S-function to the external interface program (Fig. 3.1). The programing semaphores in the data transfer interface control access to the shared memory locations ‘Publish’ and ‘Update’ and ensure reliable and orderly communication between the Simulink simulation models and the interface program.

The data transfer interface program communicates with the LOBO NCS data broker, which maintains a master record of the calculated values of the state variables by the Simulink pressurizer model and the generated control signal by the pressurizer’s PLCs. These PLCs control the system pressure and the water level in the pressurizer (Fig. 3.1). The data broker transmits the appropriate state variables to each PLC and reads back and updates the returning control signals transmitted back to the Simulink model through the data transfer interface. Separate process threads manage communication with each PLC and transmit the values of the state variables to and from the main data broker thread. The multithreaded data broker and the communication interface programs run on the same server node as the Simulink model and the data transfer interface (Fig. 3.1). In the performed test, PLCs connected either as an emulated controller in a VM or as physical hardware connected to the test network as hardware-in-the-loop.

The ManiPIO program in the LOBO NCS platform is used to introduce simulated FDIAs

targeting the inputs and outputs of the PLCs in the Ethernet testing network (El-Genk and Schriener 2022). The FDIAs function by writing to the Modbus registers to hold or falsify the values of the input state variables and the output control signals to and from the PLCs.

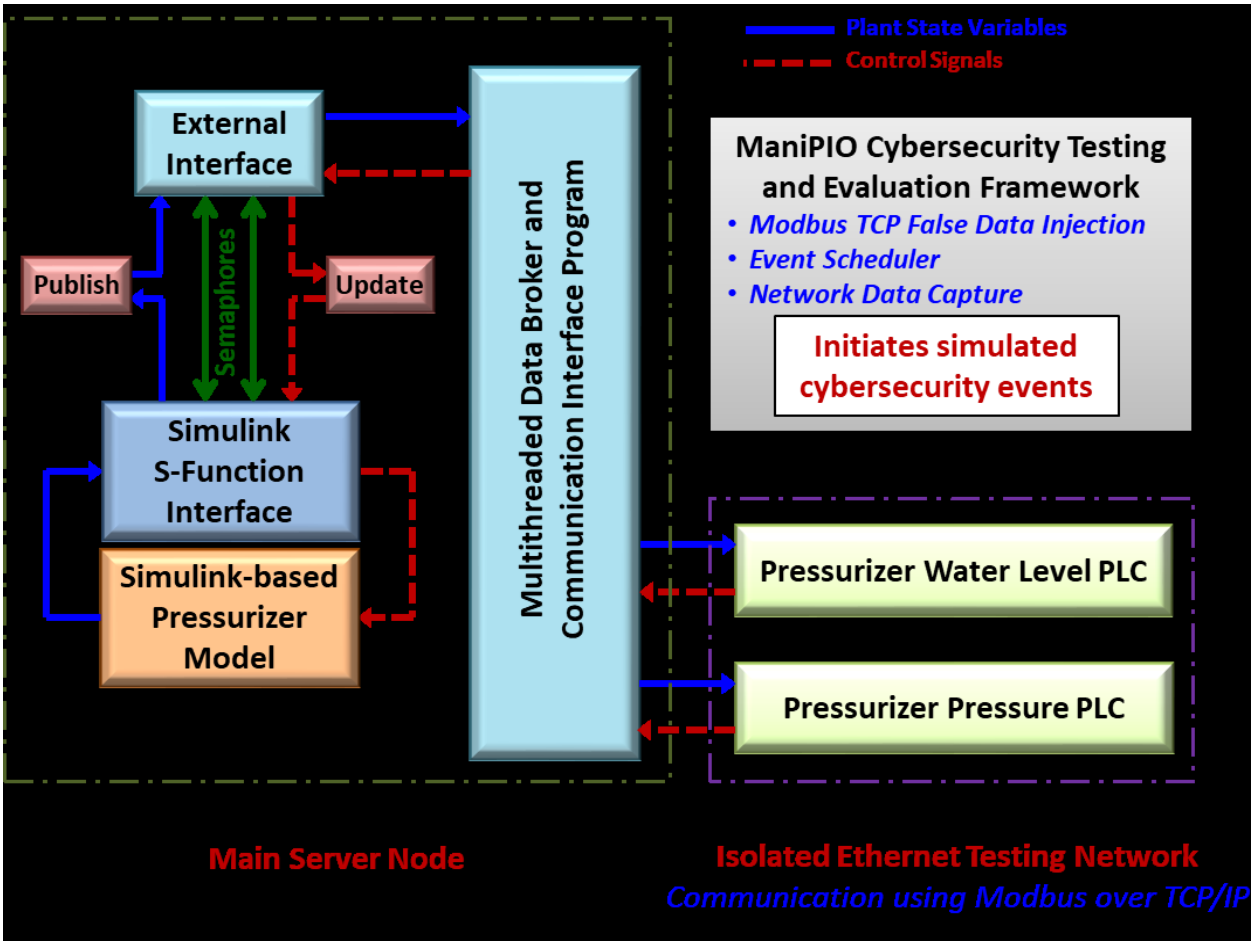

Fig. 3.1. A schematic of the LOBO NCS platform with ManiPIO for simulating FDIAs (El-Genk and Schriener 2022; El-Genk et al. 2021).

3.2. Pressurizer Model and Pressure PLC

The validated physics-based transient model of the pressurizer (Fig. 3.2) comprises three regions: an upper region of saturated vapor, a middle region of saturated liquid, and a lower surge region of subcooled liquid (El-Genk, Altamimi, and Schriener 2021; Schriener and El-Genk 2021). It accounts for the different physical processes taking place within the pressurizer. These include rainout, surface condensation on the spray water droplets, wall condensation, and flash evaporation into the saturated vapor region (Fig. 3.2). The lower region of the pressurizer accommodates the water that enters following a surge-in from the hot leg of the primary loops. Energy supplied by the immersed heaters in this region raises the enthalpy of the subcooled liquid to saturation as it merges into the middle region of the pressurizer. The pressurizer model assumes the same pressure in all three regions and solves the coupled mass and energy

conservation equations in these regions to calculate the system pressure and the water level in the pressurizer during nominal and transient operations. The model also accounts for the changes in the fluid properties in all three regions of the pressurizer as functions of pressure and temperature throughout the transient simulation (International Association for the Properties of Water and Steam 2012). More details on the formulation and validation of the pressurizer model can be found elsewhere (El-Genk, Altamimi, and Schriener 2021; Schriener and El-Genk 2021).

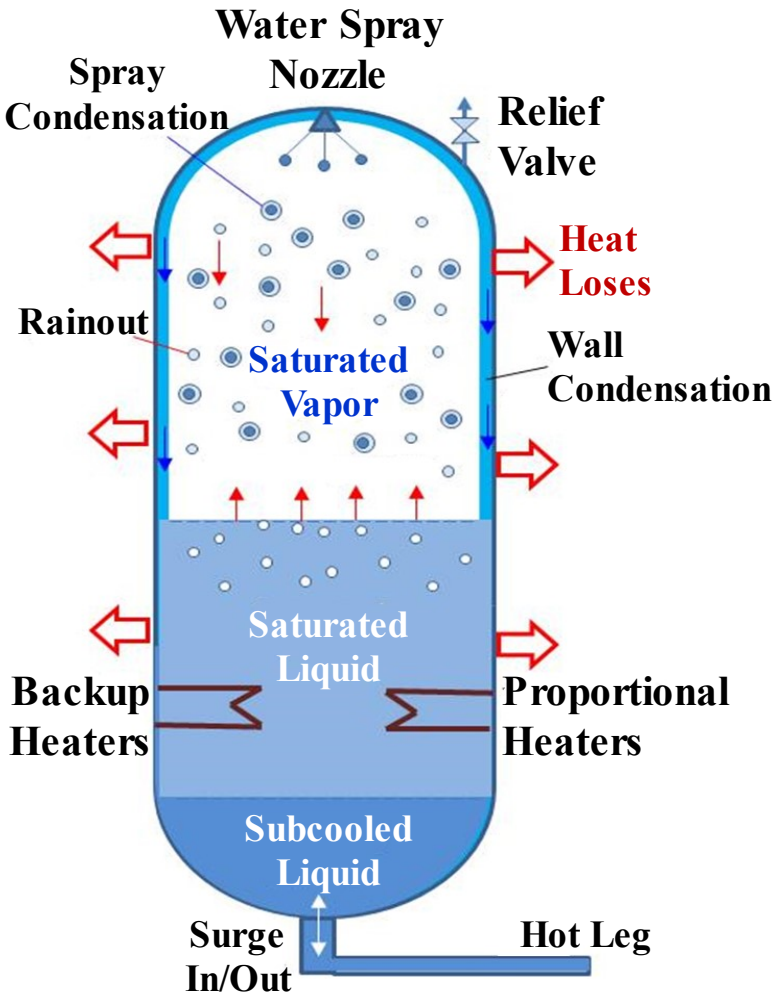

Fig. 3.2. A sketch of the pressurizer model showing various regions and physics processes (El-Genk, Altamimi, and Schriener 2021).

During the simulated FDIAs in this section the pressurizer Simulink model runs as a stand-alone program decoupled from the integrated PWR plant model. The water temperatures in the hot and cold legs of the plant specified in the input to the pressurizer model and as functions of time in simulated transients. The thermophysical properties of the surge-in water into the pressurizer evaluated at the hot-leg temperature and those of the subcooled water to the spray nozzle at the top of the pressurizer are evaluated at the cold leg temperature. This nozzle sprays

water droplets into the vapor region of the pressurizer when there is an increase in system pressure (Fig. 3.2). The pressure relief valve opens intermittently when the pressure continues to increase past a preprogramed pressure setpoint, despite the water spray, to help restore the system pressure to nominal value.

Table 3.1. Design and operating parameters of a PWR pressurizer in present analyses.

Parameter	Value	Parameter	Value
Nominal pressure, MPa	15.5	Max. water spray rate, m^3/s	0.0443
Total internal volume, m^3	59.46	Proportional heaters power, kW	370
Nominal water fill, m^3	28.3	Backup heater power, kW	1,230
Wall inner diameter, m	2.54	Relief valve setpoint, MPa	17.237
Wall outer diameter, m	2.794	Spray upper setpoint, MPa	15.858
Overall height, m	12.776	Spray lower setpoint, MPa	15.686
Strait section height, m	9.98	Proportional heaters upper setpoint, MPa	15.686
End dome radius, m	1.397	Proportional heaters lower setpoint, MPa	15.340
Surge line length, m	25.39	Backup heater on setpoint, MPa	15.168
Surge line mean dia., m	0.4572	Backup heater off setpoint, MPa	15.340

The pressure PLC regulates the system pressure by controlling the rate of water spray, the electrical power to the proportional and backup heaters (Fig. 3.3), and the opening and closing of the pressure relief valve. The PLC receives the calculated system pressure by the Simulink model of the pressurizer via the LOBO NCS data transfer and communication interfaces. It transmits back control signals to adjust the various functions in the pressurizer. The calculated system pressure is compared within the PLC's logic to the preprogramed pressure setpoints for the various control functions (Fig. 3.3, Table 3.1).

Figure 3.3 shows the responses of the programed controllers for the spray rate, and proportional and backup heaters with changes in the system pressure. When the system pressure increases past the pre-programed high setpoint of 15.686 MPa, the spray nozzle increases the rate of water spray into the vapor region of the pressurizer, proportionate to the increase in pressure until the nozzle is fully open when the system pressure reaches 15.858 MPa (Fig. 3.3). The rate of water spray in the model reaches 0.0443 m^3/s when the valve is fully open. Conversely, when the pressure decreases below the low setpoint of 15.686 MPa for the proportional heaters increase the electrical power to the heaters commensurate with the decrease in pressure until reaching 15.340 MPa (Fig. 3.3). This pressure corresponds to the reactor steady state nominal thermal power (Table 3.1). The maximum power to the proportional heaters is set at 370 kW_{th}.

With the system pressure continuing to decrease below the low pressure setpoint, the PLC

turns the immersed backup heaters on to increase flash evaporation into the top vapor region of the pressurizer (Fig. 3.2) and helps restore the system pressure. The immersed backup heaters in the lower region of the pressurizer supply 1.23 kW_{th} and unlike the proportional heaters, operate in a binary setting of either fully on or fully off. The backup heaters programed to turn on when the system pressure decreases below the setpoint of 15.168 MPa and remains on until the pressure increases above the upper setpoint of 15.340 MPa (Table 3.1 and Fig. 3.3). The pressure PLC control logic in the LOBO NCS (Fig. 3.3) applies to both the emulated and commercial hardware PLCs investigated in the present analyses. Both PLCs programed with identical logic and evaluated to ensure the same responses during nominal operation

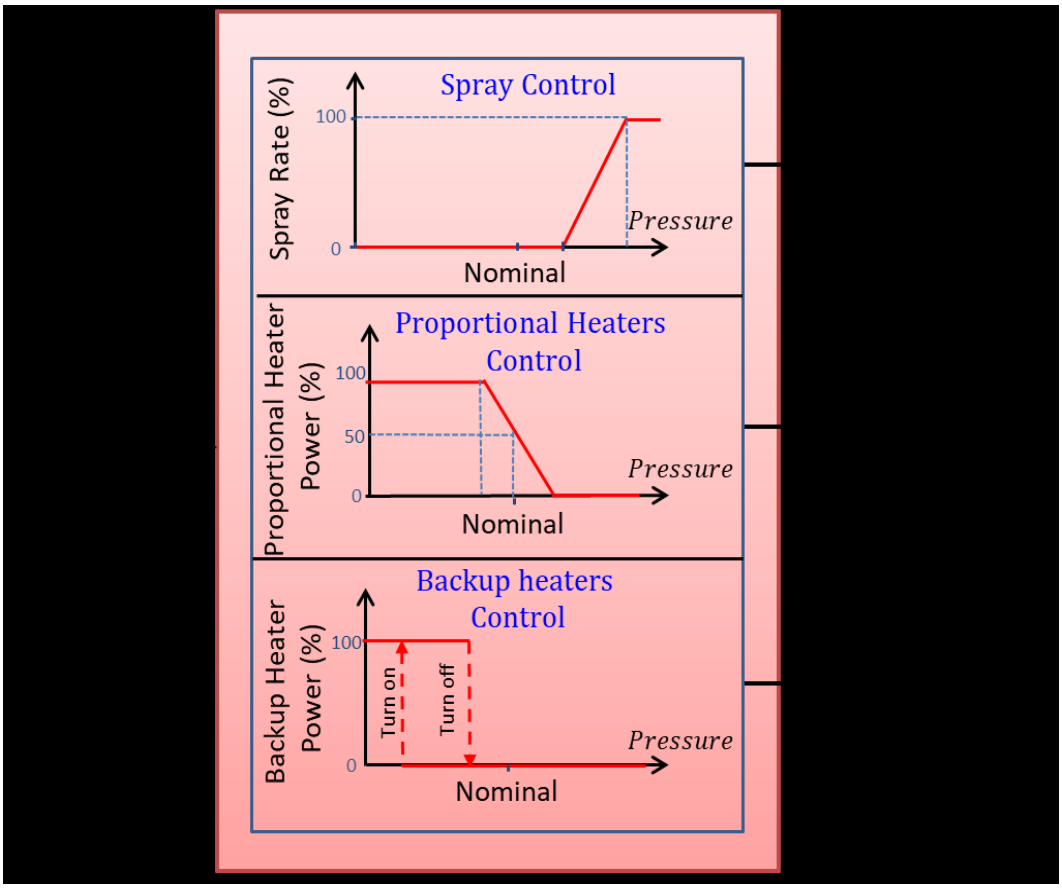

Fig. 3.3. Control functions of the pressurizer's pressure PLCs for the spray nozzle, and the proportional and backup heaters (El-Genk, Altamimi, and Schriener 2021; Schriener and El-Genk 2021).

The emulated PLC uses an open-source architecture using the OpenPLC runtime software (Alves and Morris 2018) within a Raspian OS virtual machine. It compiles and runs programs written in the IEC 61131-3 standard structured text PLC programming language (Alves and Morris 2018). The virtual machine uses the VMWare virtualization software (VMware 2019) on a Windows 10 server computer. Communication with the OpenPLC runtime is accomplished using the Modbus TCP protocol. The other PLC implemented in the present simulations is a

commercial Allen-Bradley (A-B) Micrologix L33ER PLC with ProSoft MVI69E-MBTCP module to support Modbus TCP communication (Schriener and El-Genk 2021). The ProSoft module interfaces with the Allen-Bradley's processor module through the PLC's backplane. The two PLCs are both configured with one Modbus holding register for the system pressure value, and four holding registers for the control signals for the various pressurizer functions (Fig. 3.3).

The data broker and the communication interface write the pressure generated by the pressurizer's Simulink model to the input register and return the control signals from the PLCs in the four output registers to the pressurizer model (Fig. 3.1). In a network topology the data broker and communication interface are analogous to networked sensors or RTUs sending data digitally to the PLC and receiving the values of the output digital control signals. The Simulink pressurizer model uses a simulation timestep of 10 ms while the input scan time for the emulated PLC with OpenPLC and the Allen-Bradley physical PLC are varied from 1.2 ms to 450 ms. The testing network connects the emulated PLC and the server running the virtual machine. The Allen-Bradley Micrologix PLC connected through the attached ProSoft module to the LOBO NCS platform as hardware-in-the-loop. The ManiPIO program runs on a separate computer from the emulated PLC representing a cyber-compromised device on the network that sends false Modbus TCP traffic to the pressure PLC.

The next section presents and compares the results of the linked Simulink model of the pressurizer to both the emulated and the physical Allen-Bradley pressure PLCs during a simulated transient involving sequential surge-in and surge-out events from a representative PWR plant. These results are of normal operation and while the pressure PLC subjected to simulated Modbus FDIAs generated by the ManiPIO program (Fig. 3.1). The compared results are of the pressurizer with both the emulated and commercial hardware PLCs under simulated FDIAs.

3.3. Simulated Surge-in and Surge-out Transients

The emulated and the commercial Allen-Bradley pressure PLCs linked to the pressurizer's physics-based Simulink model in the LOBO NCS platform and used to model a simulated surge-in and surge-out transient for input scan times between 2 - 20 ms. The results in Figs. 3.4 and 3.5 compare the responses of the two PLCs for input scan times of 2 and 5 ms. The simulated transient starts at t = 0 s with the pressurizer at a steady state system pressure of 15.686 MPa, water level in the pressurizer of 4.22 m, and hot leg water temperature of 564.8 K. The surge-in of water from the hot leg into the pressurizer starts at t = 100 s (point 1 in Fig. 3.4a) and its rate increases linearly to of 25 kg/s (point 2 in Fig. 3.4a) over a period of 50 s. The surge-in rate is then held steady at 25 kg/s for 100 s (point 3 in Fig. 3.4a) before decreasing linearly to 0 kg/s over an additional a period of 50 s (point 4 in Fig. 3.4a). The surge-in raises the water level (Fig. 3.4a) and compresses the vapor in the upper region of the pressurizer (Fig. 3.2), which increases the system pressure (Fig 3.4b). In response to the pressure increase the PLCs increase the spray rate of water droplets into the upper vapor region of the pressurizer (Fig. 3.2). The vapor condensation onto the water droplets and on the inner surface of the pressurizer wall decreases the system pressure (Fig. 3.4b).

During the progression of the surge-in event the pressure PLCs increase the spray rate of water droplets, which peaks at 31 kg/s at t = 235 s. After that, the spray rate decreases to zero as the system pressure drops below the setpoint for opening the spray nozzle (Fig. 3.5a). The surge-in of water from the hot-leg and the injected water through the spray nozzle increase the water level in the pressurizer. The sequential surge-out of water from the pressurizer into the hot leg begins at t = 400 s of the simulated transient (point 5 in Fig. 3.4a) and increases linearly from 0 kg/s to 25 kg/s over a period of 50 s (point 6 in Fig. 3.4a). It then holds constant at 25kg/s for 100 s (point 7 in Fig. 3.4a), before decreasing linearly to 0 kg/s over a period of 50 s (point 8 in Fig. 3.4a). The surge-out event decreases the water level in the pressurizer (Fig. 3.4a), and the associated vapor expansion in the top region of the pressurizer (Fig. 3.2) decrease the system pressure (Fig. 3.4b).

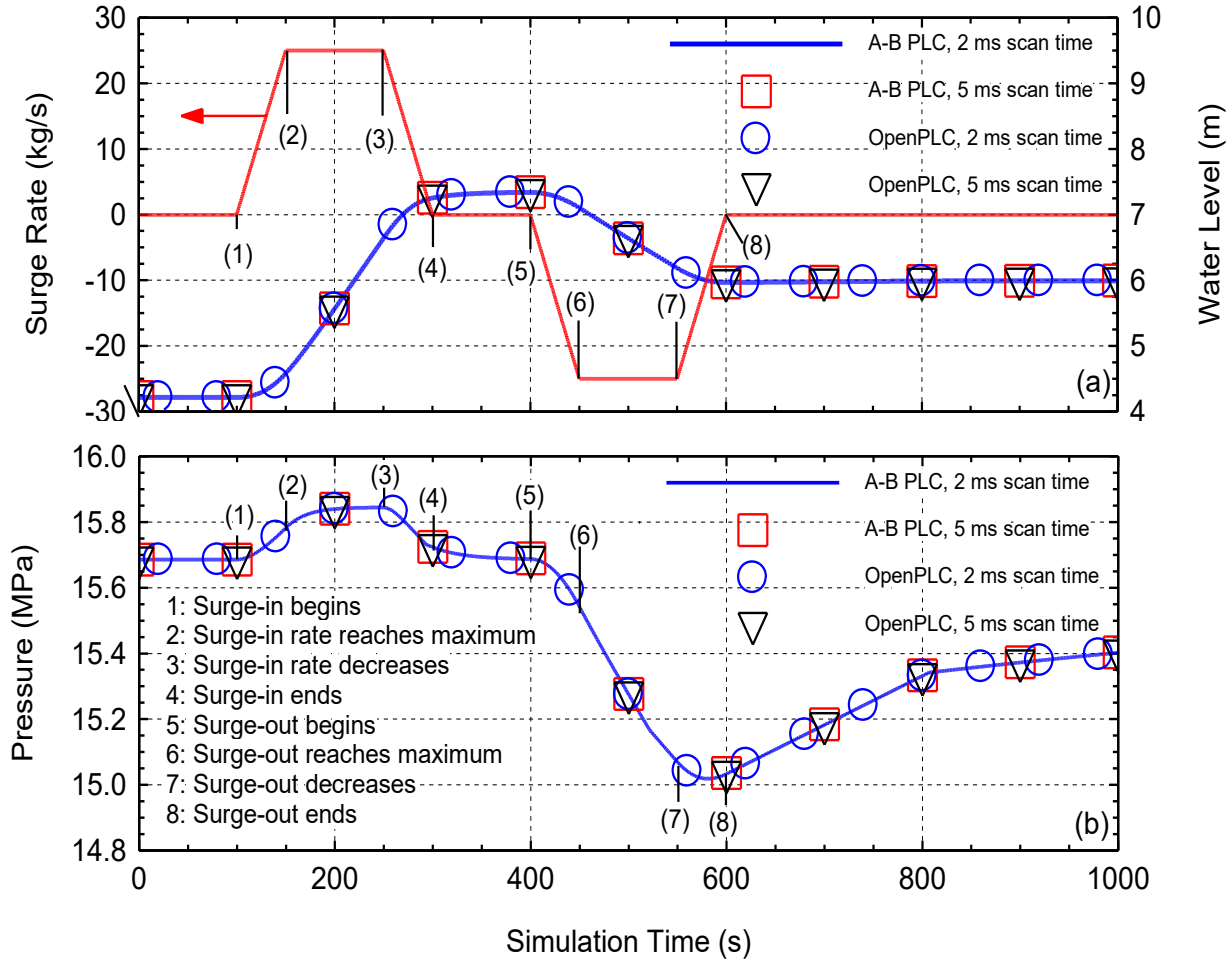

Fig. 3.4. Comparison of the responses of the Allen-Bradley hardware and the emulated PLCs with input scan times of 2 and 5 ms during a sequential surge-in and surge-out events.

When the pressure drops below the setpoint for the proportional heaters, the PLCs increase the electrical power to the heaters which helps increase the rate of flash evaporation into the top vapor region (Fig. 3.2) and hence, the system pressure (Figs. 3.4, 3.5). Nonetheless, the system

pressure continues to decrease because of the rapid surge-out of water from the pressurizer into the hot leg even after the power of the proportional heaters reaches a maximum of 370 kW_{th} (Fig. 3.5b) (Table 3.1). The backup heaters switch on when the pressure reaches the corresponding pressure setpoint and supply an additional 1,230 kW_{th} to increase flash evaporation into the vapor region of the pressurizer (Fig. 3.5c). The backup heaters stay on until the increases pressure reaches the setpoint to turn them off. In the meantime, the proportional heaters stay on, slowly increasing the pressure until it reaches a new steady state level.

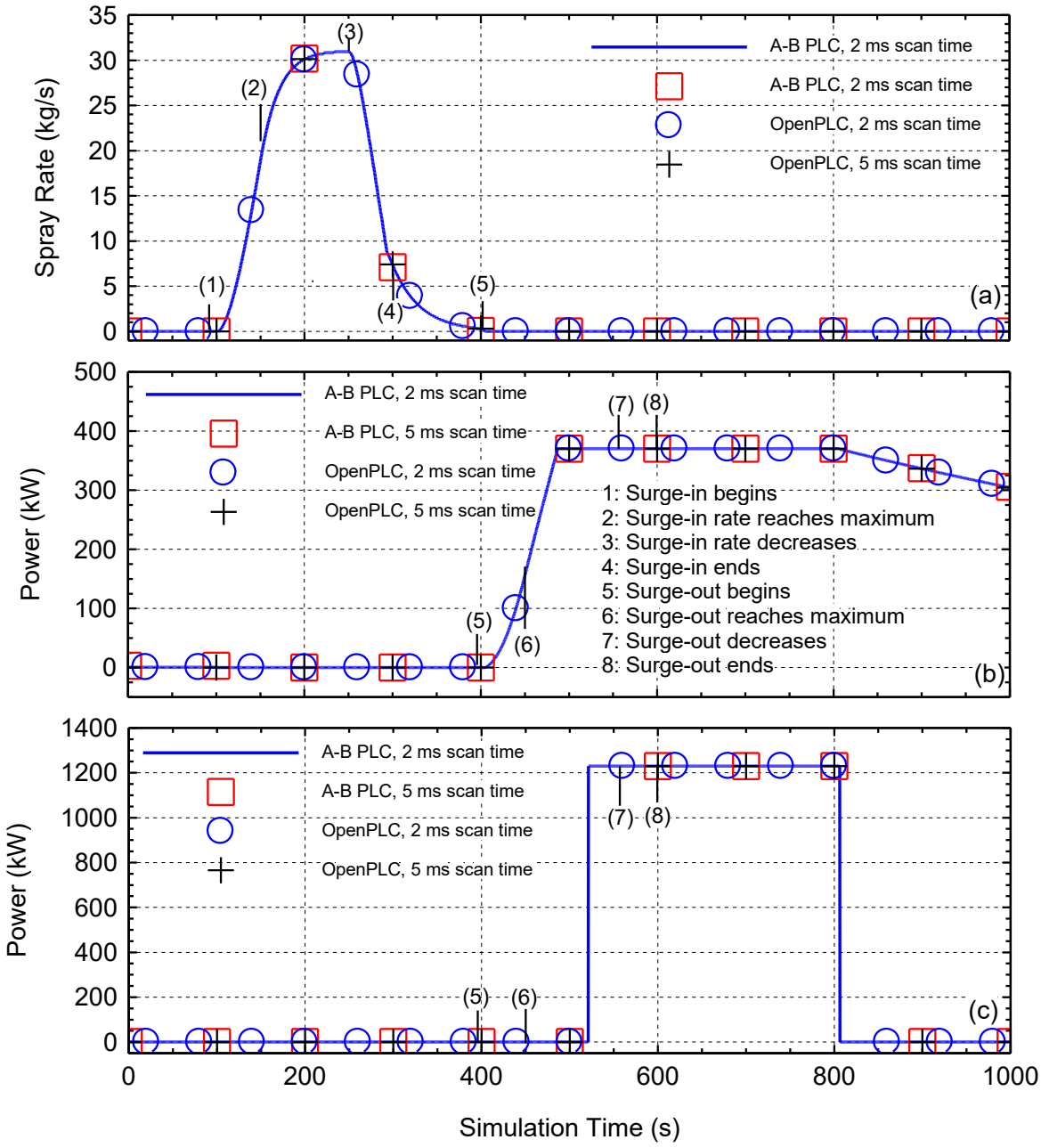

Fig. 3.5. Responses of the water spray and heaters with the Allen-Bradley hardware and the emulated PLCs with input scan times of 2 and 5 ms.

Following the end of the surge-out of water into the hot leg, the water in the pressurizer reaches a higher steady state level than the initial value at the start of the simulated transient (Fig. 3.4a). The simulated sequential surge-in and surge-out add and remove equivalent masses of water to and from the pressurizer, the difference in the water level before and after the simulated events is due to the injected water spray during the surge-in phase of the transient.

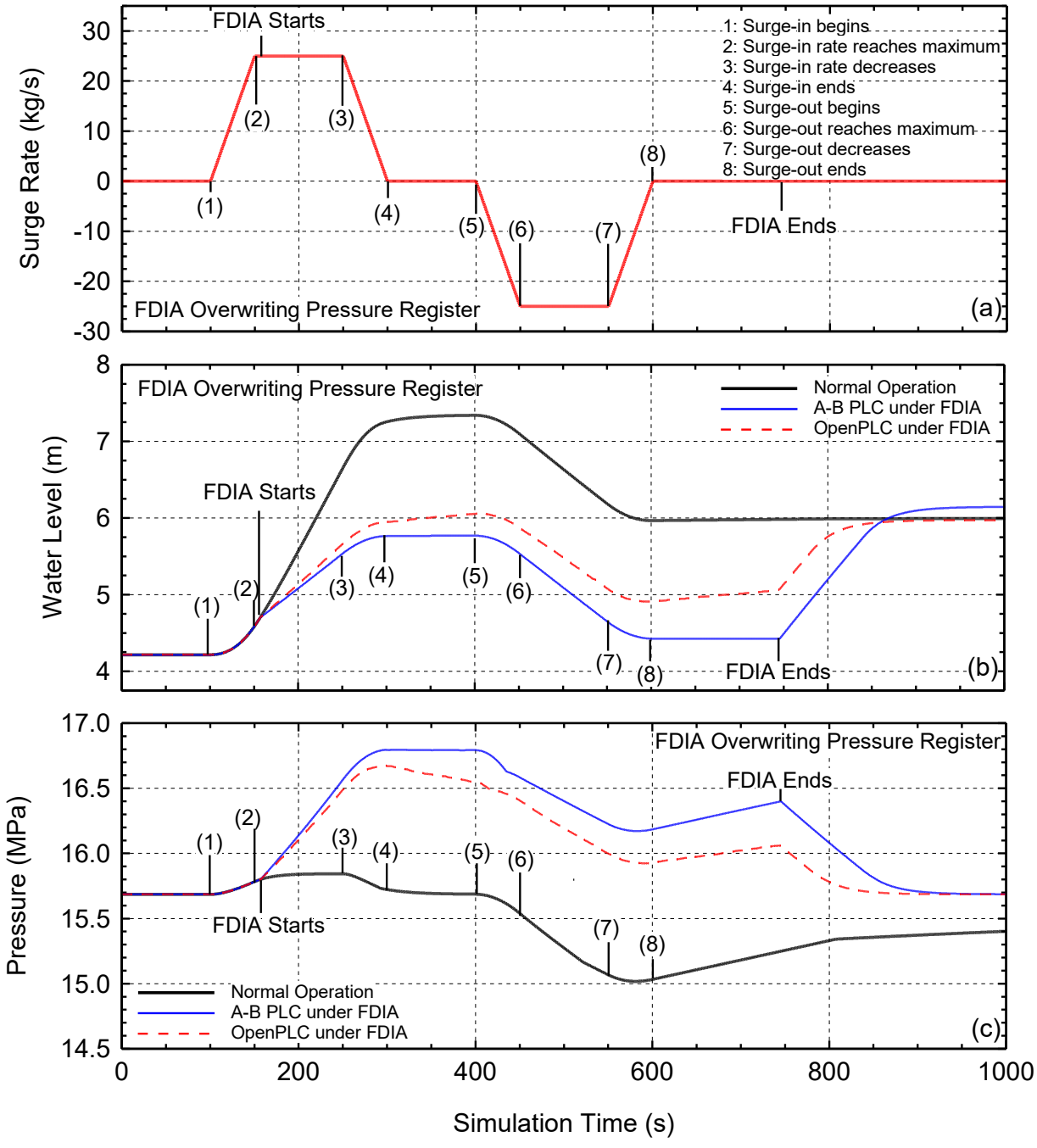

Fig. 3.6. Comparisons of responses with Allen-Bradley hardware and emulated PLCs in simulated sequential surge-in and surge-out events during normal operation and under an FDIA.

In summary, the simulated transient results in Figs. 3.4 and 3.5 show that the responses with the emulated PLC using the OpenPLC runtime in a virtual machine with 2 and 5 ms input scan times are indistinguishable from those with the commercial Allen-Bradley Micrologix PLC connected as hardware-in-the-loop (Figs. 3.4, 3.5a-c). This agreement holds true for higher input scan times of 10 and 20 ms. These results confirm that during nominal operation transients the emulated and hardware PLCs linked to the pressurizer Simulink model on the LOBO NCS platform display the same system response.

3.3.1. Responses of PLCs while under a False Data Injection Attack

The results presented in this subsection are of the responses of the emulated and commercial hardware PLCs subjected to a simulated Modbus FDIA sending a false value for the system pressure. The ManiPIO program in the LOBO NCS platform simulates a FDIA on the holding register of the PLC by sending a false input pressure value to the PLC (Fig. 3.3). The Simulink pressurizer model used to simulate the same sequential surge-in and surge-out transients described earlier (Section 3.3). The FDIA starts shortly after the surge-in rate reaches its highest value. It writes a false pressure of 15.0 MPa to the Modbus input holding register associated with the system pressure. At this pressure, the logic programming of the PLCs should signal to the proportional heaters to operate at their maximum power and to the backup heaters to turn on (Table 3.1). The FDIA continues to overwrite the register during the durations of the simulated surge-in and surge-out events (Fig. 3.6a). When the FDIA ends, the pressure PLC returns to its normal operation state and attempts to restore the system pressure back to the correct value as defined by pre-programmed pressure setpoints (Table 3.1).

The results in Figures 3.6-3.9 for a 2 ms input scan time show that the FDIA successfully alters responses of the PLCs compared to nominal operation. Overwriting the pressure holding register causes the logic programing of the PLCs to falsify the control signals to the water spray nozzle and to the proportional and backup heaters to change the water level and system pressure in the pressurizer. Nominally the two PLCs produce identical responses to the water surge-in by increasing the spray rate of water droplets into the vapor region of the pressurizer (Fig. 3.2), commensurate with the increase in system pressure (Fig. 3.6c). The surge-in and the spray rate initially increase the water level in the pressurizer (Fig. 3.6b). FDIA starts injecting a false low-pressure value to the register of the PLCs, which responds by prematurely reducing the spray rate to 0 kg/s (Fig. 3.7b-c), raising the electrical power to the proportional to its maximum, and turning on the backup heaters (Fig. 3.8b-c and 3.9b-c). The manipulation of the PLCs causes a significant increase in the system pressure compared to normal (Fig. 3.6c). The system pressure peaks at 16.8 MPa with the Allen-Bradley PLC, and 16.7 MPa with the emulated PLC, compared to only 15.8 MPa during nominal operation without an FDIA.

The combined effect of the increased pressure and reduced rate of the water entering the pressurizer when the PLCs are under an FDIA slows the increase in the water level in the pressurizer, compared to normal. During nominal operation, the water level peaks at 7.34 m, compared to 5.94 m and 5.78 m with the Allen-Bradley PLC and the emulated PLC when targeted by the simulated FDIA (Fig. 3.6b). During the FDIA, the ManiPIO program repeatedly

overwrites the input register for the system pressure to force the PLC to falsely maintain the proportional heaters at their maximum power and to turn on the backup heaters. At the same time, the LOBO NCS data broker and communication program is competing to write the actual pressure value to the same Modbus holding register.

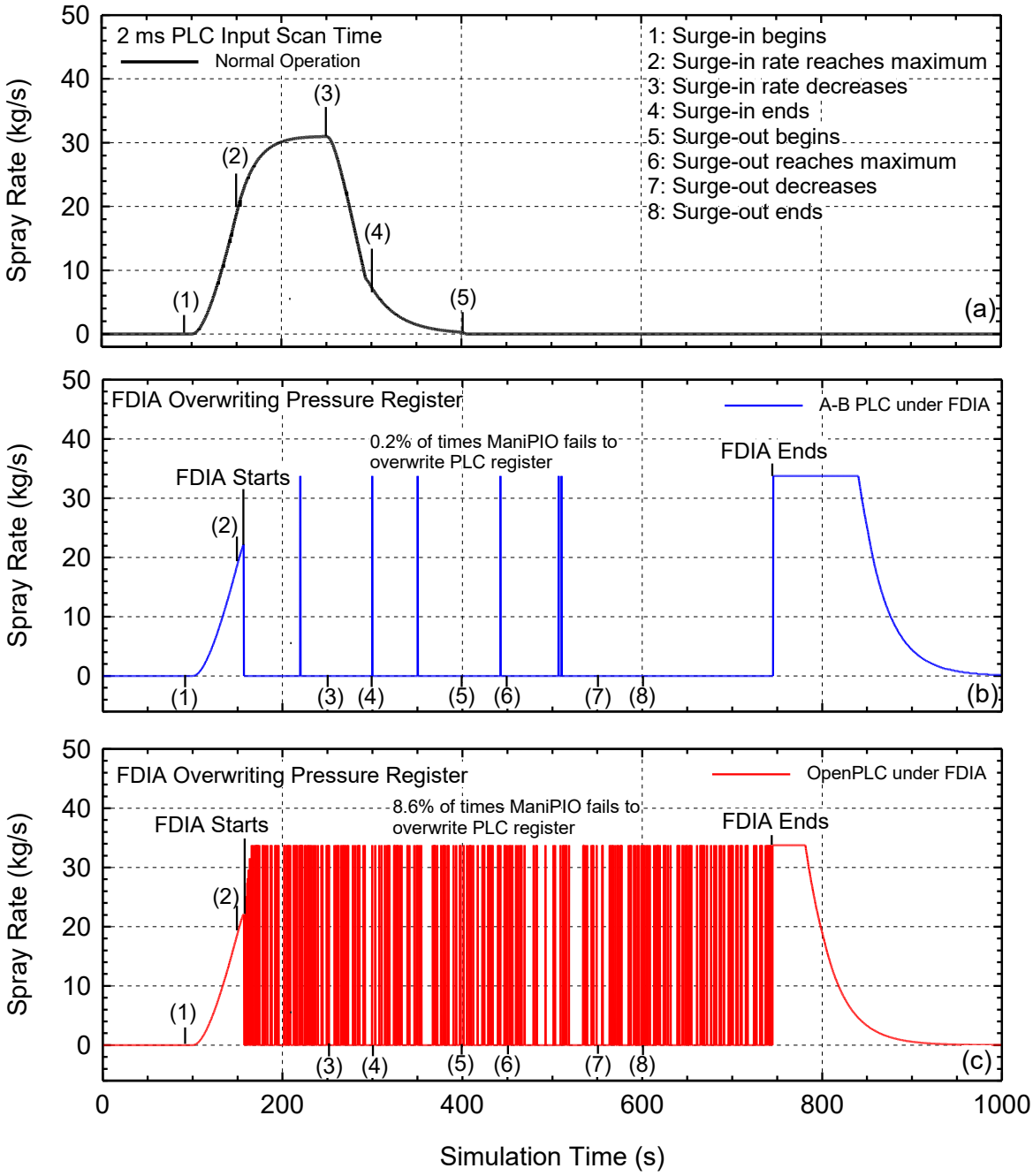

Fig. 3.7. Responses of the water spray function with the PLCs of Allen-Bradley hardware and that emulated with OpenPLC, nominally and while under an FDIA.

At few instances during the attack, the PLC manages to receive the real pressure despite the attempts by the ManiPIO program to overwrite it. This response for the Allen-Bradley PLC in Figs. 3.7b, 3.8b, and 3.9b and for the emulated PLC with Open PLC in Figs. 3.7c, 3.8c, and 3.9c are as series of vertical spikes. In response to the higher pressure these PLCs react by turning the heaters off (Figs. 3.8b-c and 3.9b-c) and opening the spray nozzle (Figs. 3.7b-c). When the ManiPIO once again successfully overwrites the register, the PLCs closes the spray nozzle and turns the proportional and backup heaters back on (Figs. 3.8b-c and 3.9b-c). The results for the emulated PLC show similar frequent inconsistent overwrite events (Figs. 3.7c, 3.8c, 3.9c).

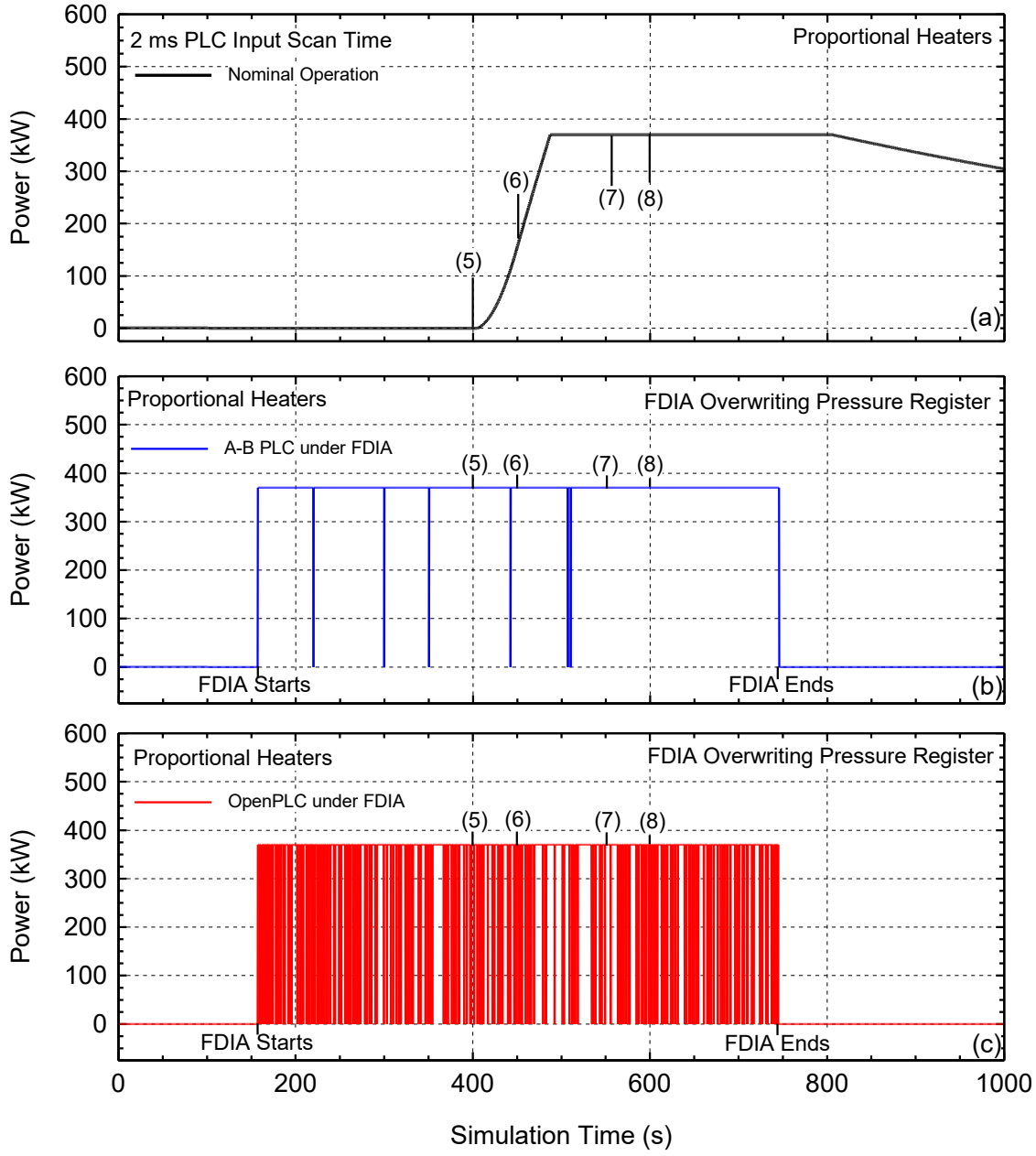

Fig. 3.8. Response of the proportional heaters control function with Allen-Bradley hardware and the emulated PLCs, nominally and under an FDIA.

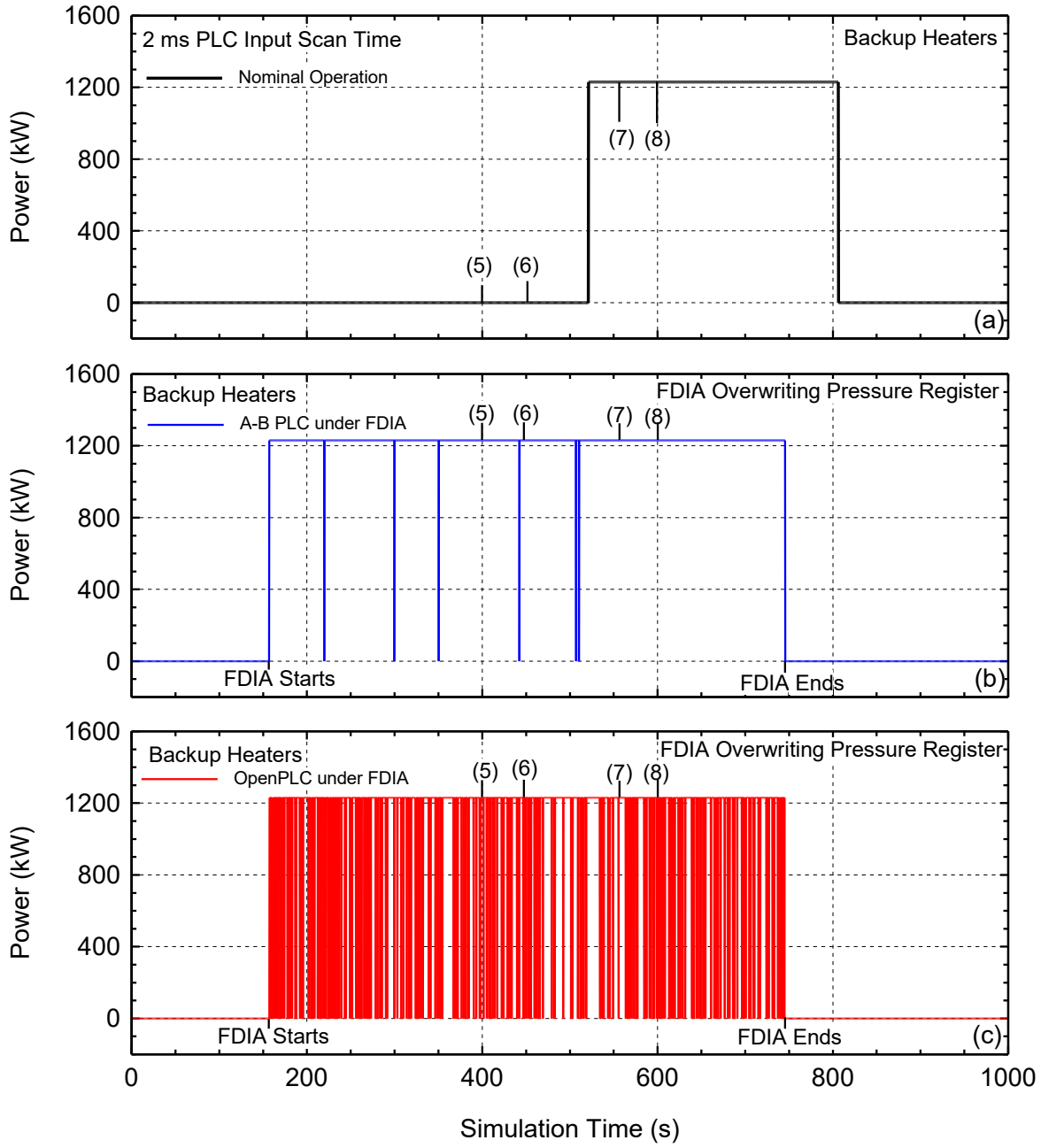

Fig. 3.9. Backup heater control functions using Allen-Bradley hardware PLC and the emulated PLC during normal operation and under an FDIA.

After the FDIA ends at t = 747s in the simulated surge-in and surge-out transient, the pressure PLC returns to normal operation and the controller shuts off the proportional and backup heaters (Figs. 3.8b-c and 3.9b-c) in response to the true high system pressure and increases the rate of the water spray to reduce the pressure (Figs. 3.7b-c). The system pressure decreases slowly due to the condensation of vapor onto the surface of the injected spray water droplets and eventually levels off at a lower steady state value (Fig. 3.6c). The simulated FDIA by the ManiPIO program

competes with the communication interface with both sending Modbus TCP write requests to the holding register of the PLCs. This race condition determines the last write request accepted prior to the PLCs reading the stored register values at the beginning of their scan cycle.

The LOBO NCS communication program sends the new state variable values received from the pressurizer Simulink model once every 10 ms. In contrast, the determined time between Modbus write requests by the ManiPIO program averages 1.10 ms. In the results presented in Figs. 3.6-3.9 the PLCs' scan time of 2 ms is ~1/9 the period of the data interface communication but is only ~1/2 that of the ManiPIO communication. These time differences altered the responses of the two PLCs when subjected to the simulated FDIA.

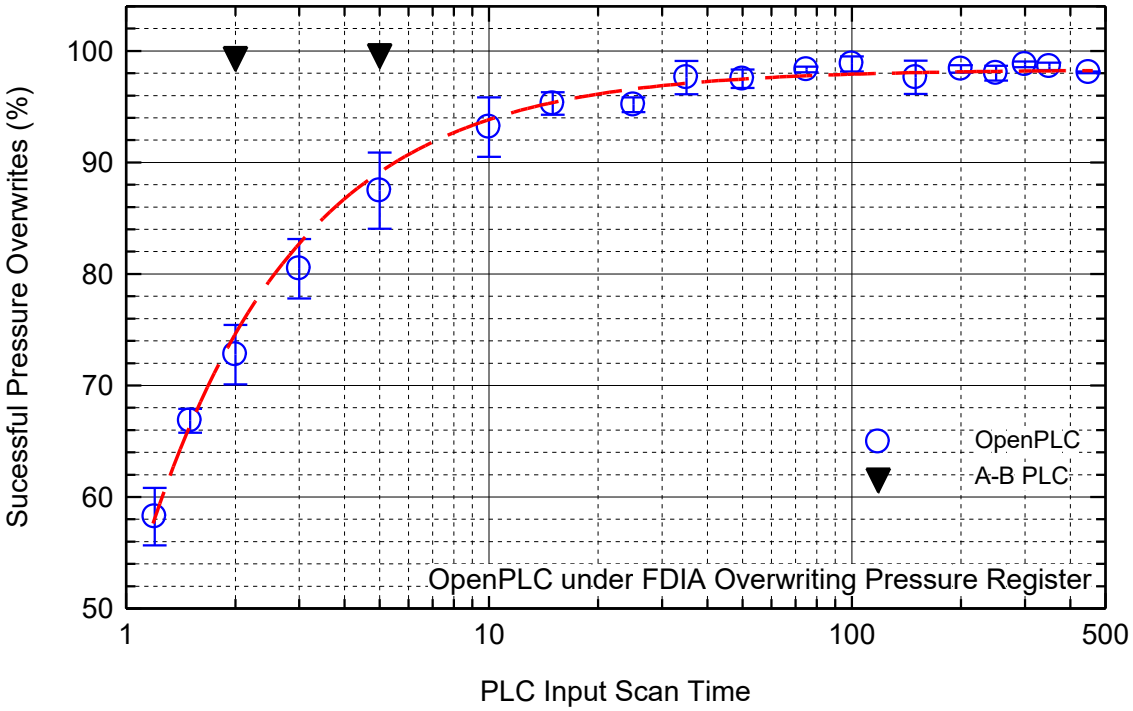

Fig. 3.10. Effect of increasing the PLC scan time on the success percentage of the Modbus holding register overwrites for the emulated PLC during simulated FDIA targeting the system pressure.

The emulated PLC with OpenPLC relies on an emulated network connection that passes through the network adapter of the server node running the virtual machines. The OpenPLC runtime has a software network communication module that manages Modbus TCP communication on its open comm port (Alves and Morris 2018). On the other hand, the Allen-Bradley PLC uses the attached ProSoft module to manage the Modbus TCP communication with the data broker and ManiPIO (Allen-Bradley 2013). This module passes on the values of the written state variable and control signals to and from the PLC's processor module across the backplane on regular scheduled intervals during the PLC's scan cycle (Allen-Bradley 2013).

With the independently running ProSoft module managing the incoming Modbus TCP traffic as contrasted to the internal subroutine in OpenPLC, the Allen-Bradly PLC enacts a larger

number of the periodic write requests sent by the simulated FDIA. Consequently, the response of the Allen-Bradley PLC while subject to FDIA is more consistent. On the other hand, the response of the emulated PLC is more irregular as to when it accepts and enacts the competing Modbus TCP write requests sent by the two programs. Despite the ManiPIO program sending write requests nine times more frequently than the LOBO NCS communication interface, the FDIA regularly fails to supersede the true pressure value.

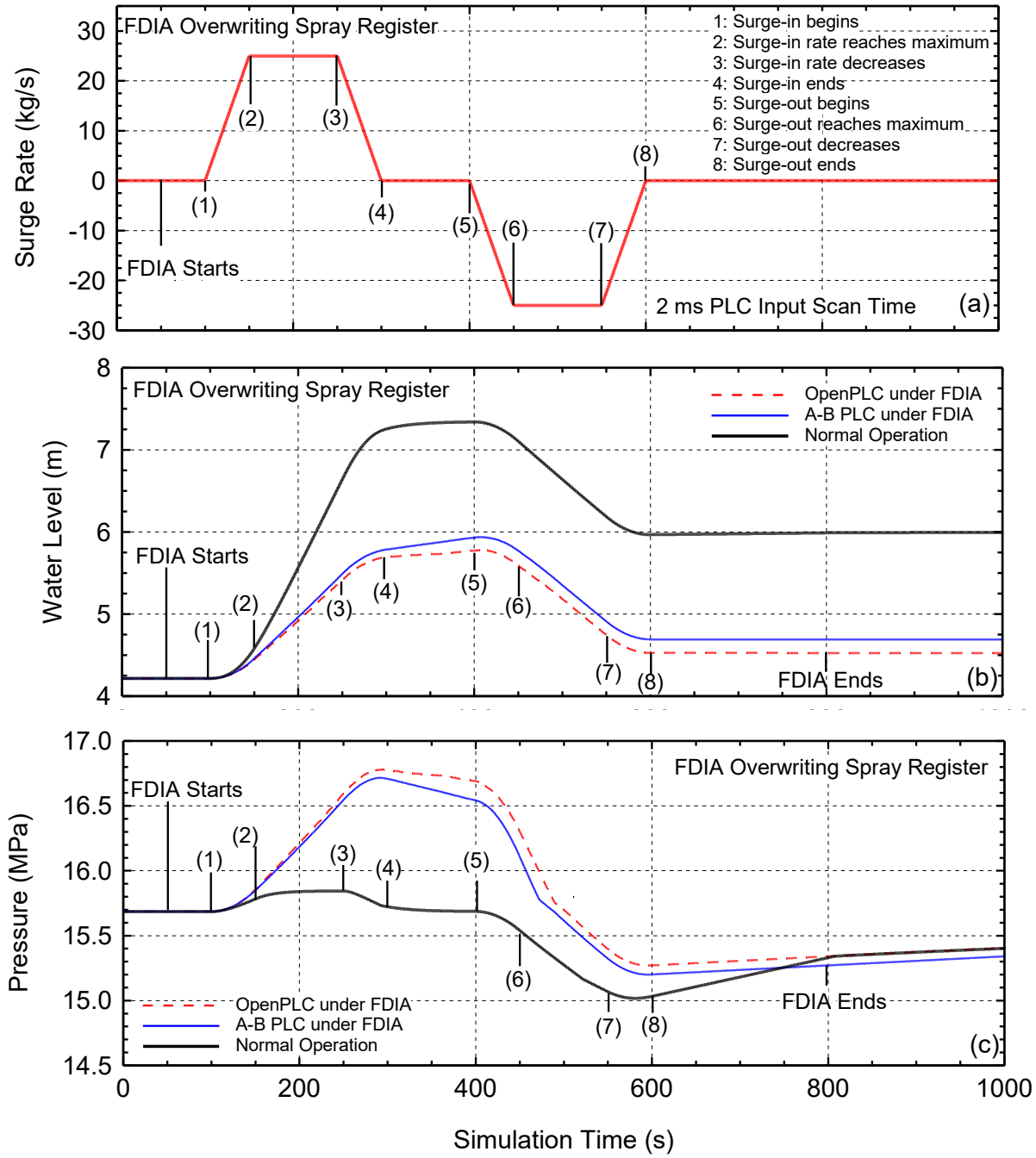

Fig. 3.11. Comparison of the responses of water spray emulated and A-B PLCs, during normal operation and under the simulated FDIA pressure.

The small difference in timing between the PLC scan time and the period between the Modbus TCP communications by ManiPIO also contributes to the irregular overwriting of the pressure holding register with the emulated PLC. To determine its effects on the success rate of overwriting the register for the OpenPLC, the simulated FDIA repeated for input scan times between 1.2 ms to 450 ms (Fig. 3.10). The scenario repeated for each scan time three times to determine the range of the successful percentage of the overwrites. As the programed scan time of the PLC increases the percentage of the successful overwrites increases. When the value of the input scan time approaches ~100 ms the ManiPIO program can successfully overwrite the pressure value > 98% of the time.

Fig. 3.10 shows the results with the Allen-Bradley PLC under the simulated FDIA with input scan times of 2 ms and 5 ms. In both cases, the FDIA succeeded in overwriting the Modbus holding register for the system pressure > 99.5% of the times. The susceptibility of the A-B PLC under the simulated FDIA continues at higher input scan times (Fig. 3.10). The long scan time provides more opportunities for the PLC to enact the Modbus TCP write requests sent by ManiPIO before it reads the register vales into the logic control program. With the communication periods for the LOBO NCS interface and the simulated FDIA not synchronized, there is a possibility that the last enacted write request by the PLC will be the true pressure value and not the false value from the FDIA. Consequently, the simulated FDIA occasionally fails to supersede the true input value even for long PLC scan times.

The presented results show the simulated FDIA successfully alters the operation of both the emulated and the commercial Allen-Bradley PLCs during most the attack period. The emulated PLC with OpenPLC is altered less than the hardware Allen-Bradley PLC (91.6% compared to 99.8%). For both PLCs, the simulated FDIA causes a significant increase in the system pressure. The success percentage of the simulated FDIA overwriting the holding register of these PLCs depends on the scan time of the PLC and increases monotonically with increased input scan time. The next subsection compares the responses of the emulated and Allen-Bradley PLCs controlling the injection of water spray droplets into the vapor region of the pressurizer (Fig. 3.6).

3.3.2. Responses of Water Spray PLCs to a FDIA of Holding Register

In the second simulated FDIA scenario the ManiPIO program overwrites the Modbus holding registers of the emulated and hardware PLCs to alter the output control signal for the water spray nozzle in the pressurizer (Fig. 3.2). This scenario investigates how the responses of the PLC differ from the previous case which targeted one of the PLC's input values. This FDIA on the pressure PLCs attempts to supersede the output value stored in the holding register of the PLC. This value updates internally based on the scan cycle time and communicated when requested to the LOBO NCS communication and data broker program.

The physics-based Simulink model of the pressurizer simulates the sequential surge-in and surge-out transient described earlier (Fig. 3.4a) with the simulated FDIA on the holding register of the PLCs for the water spray occurs 50s before the start of the surge-in event (Fig. 3.11a). The ManiPIO program writes zero water spray rate (0.0 kg/s) to keep the water spray nozzle closed to increase the system pressure during to surge-in of water from the hot leg. The simulated FDIA

lasts for a total of 800s and when it ends the emulated and the A-B hardware PLCs returns to operating normally (Figs. 3.11a-c).

Figures 3.11-3.13 compare the responses of the emulated and Allen-Bradley PLCs with input scan time of 2 ms, both during nominal operation when targeted by the simulated FDIA. During the simulated surge-in transient the FDIA disrupts the water spray control function of the PLC causing the system pressure to increase beyond the maximum of 15.8 MPa (Table 3.1, Figs. 3.11c). With the Allen-Bradley PLC under the simulated FDIA the pressure peaks at 16.7 MPa compared to 16.8 MPa with the emulated PLC (Fig. 3.11c). The FDIA suppressing the activation of the water spray decreases the water level in the pressurizer compared to nominal due to the lower rate of the water spray injection (Fig. 3.11b). With the Allen Bradley PLC the water level in the pressurizer peaks at 5.94 m, compared to 5.78 m with the emulated PLC and to 7.34 m when both PLCs are operating normally.

The water spray PLCs control the proportional and backup heaters during the simulated transient without and with the simulated FDIA targeting their holding register (Fig. 3.2). With nominal operation at the start of the surge-out from the pressurizer to the hot leg (t = 400 s), the PLCs increase the power to the proportional heaters (Fig. 3.13a). When the PLCs are under an FDIA the high system pressure at the start of the surge-out event (Fig. 3.11c) delays the activation of proportional heaters (Figs. 3.13b and c) and is higher than the setpoint for turning on the backup heaters (Fig. 3.13d and Table 3.1). The differences in the responses of the Allen-Bradley hardware PLC and the emulated PLC during the simulated FDIA targeting the water spray function (Figs. 3.11-3.13) are smaller than that observed during that targeting the system pressure (Fig. 3.7-3.9). The similar responses of the Allen-Bradly and the emulated PLCs with the FDIA is because the PLCs update their register values during their internal scan cycles on the same interval as the input scan time of two ms. In addition, the simulated FDIA is not competing with the communication interface to write to the input registers as it is in FDIA targeting the pressure input holding register.

The differences in how the two PLCs manage incoming traffic at different rates from multiple sources does not affect their overall response. However, a competition does develop for the ManiPIO program attempting to overwrite the holding register of the PLCs. The ManiPIO program is not competing with the Modbus TCP traffic from the communication interface, but instead with the PLC's scan cycle. The simulated FDIA sends repeated write requests to the holding register of the PLCs controlling the water spray rate to secure a false value in place before the communication interface sends the Modbus TCP read request for the stored value.

The analysis of the simulated FDIA on the water spray PLCs is repeated for different input scan times to determine the effect the FDIA success of overwriting the rate of water spray (Fig. 3.14). The emulated PLCs used input scan times between two ms and 450 ms, with the Allen-Bradley PLC used input scan times of 2 and 5 ms. The results in Fig. 3.14 show that increasing the input scan time for the emulated PLC increases the success percentage the simulated FDIA to overwrite the output register value.

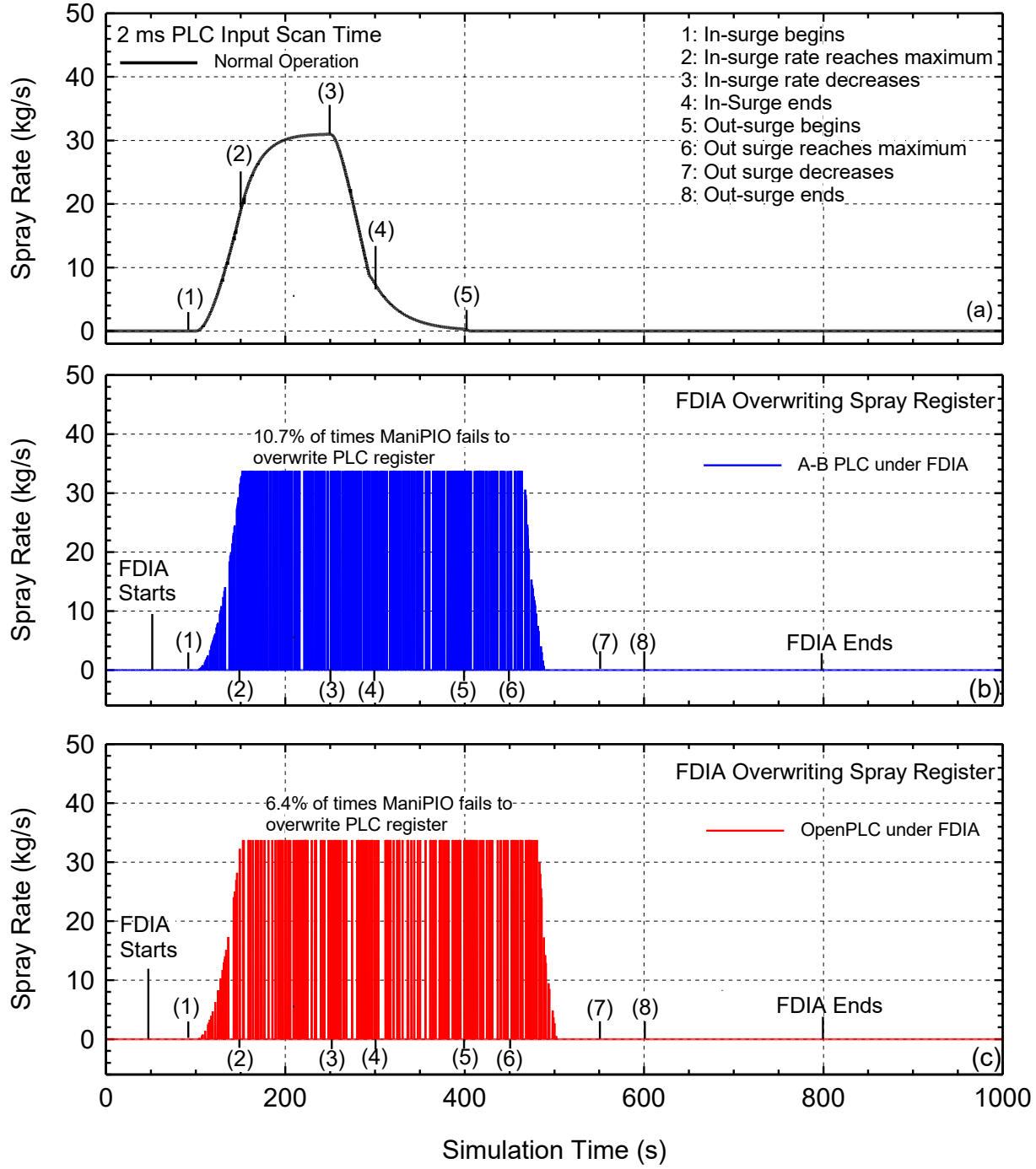

Fig. 3.12. Responses of the Allen-Bradley hardware and the emulated PLCs during normal operation and while under an FDIA targeting the water spray control.

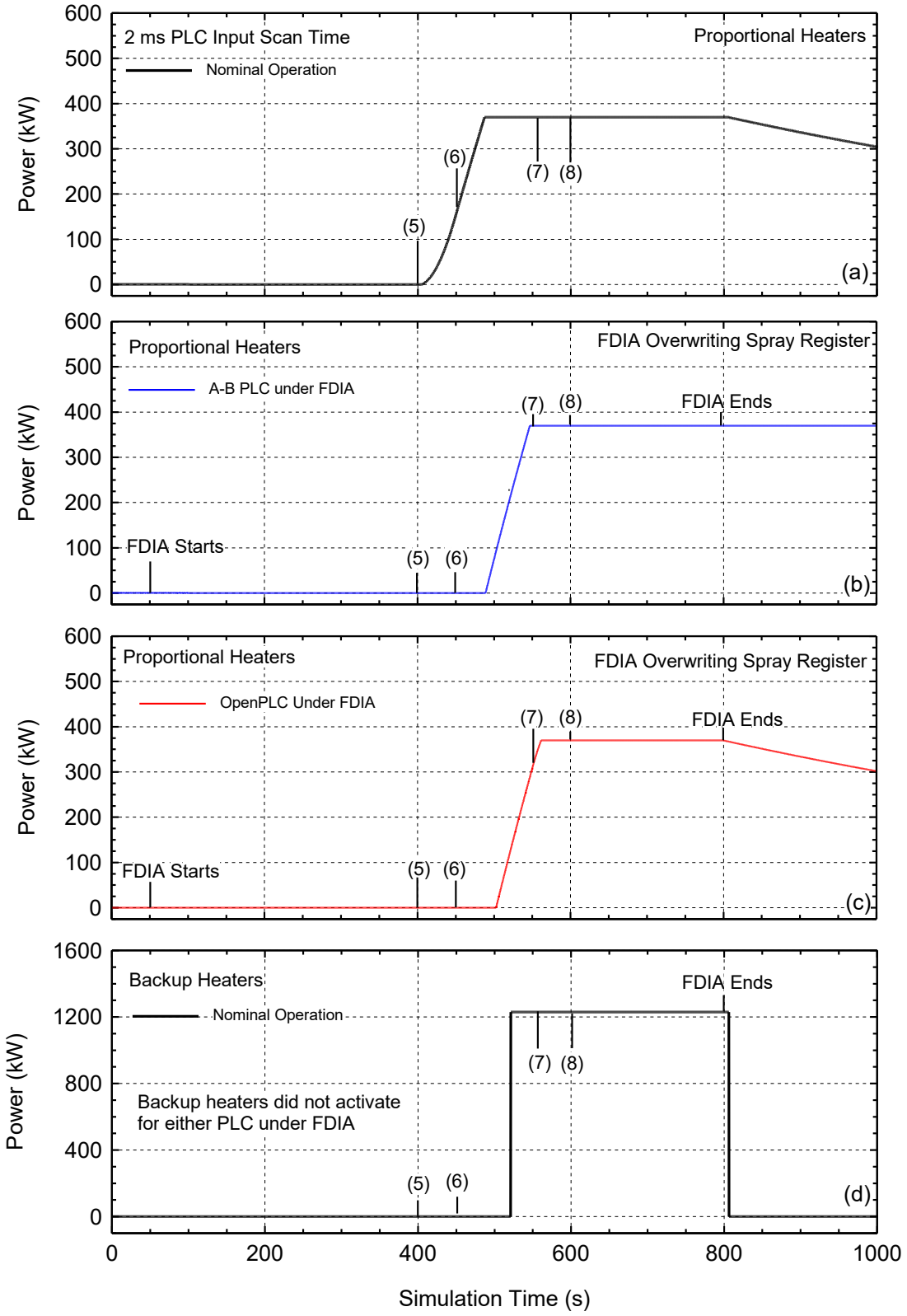

Fig. 3.13. Responses of proportional heaters controlled by the Allen-Bradley hardware PLC and the emulated OpenPLC with and without an FDIA targeting water spray control.

Increasing the PLC input scan time increases the period between the successive overwrites by the PLC to the output holding register. The period of the ManiPIO Modbus TCP writes requests (~1.1 ms on average) is much shorter than that for the read requests sent by the communication interface (10 ms). When the input scan time of the PLC increases beyond~ 50 ms the simulated FDIA successfully supersede the true spray rate value > 99% of the time. The obtained results with the Allen-Bradley hardware PLC suggest that it is consistently more susceptible to the simulated FDIA than the emulated PLC. This is consistent with the trend observed for the FDIA targeting the pressure input holding register.

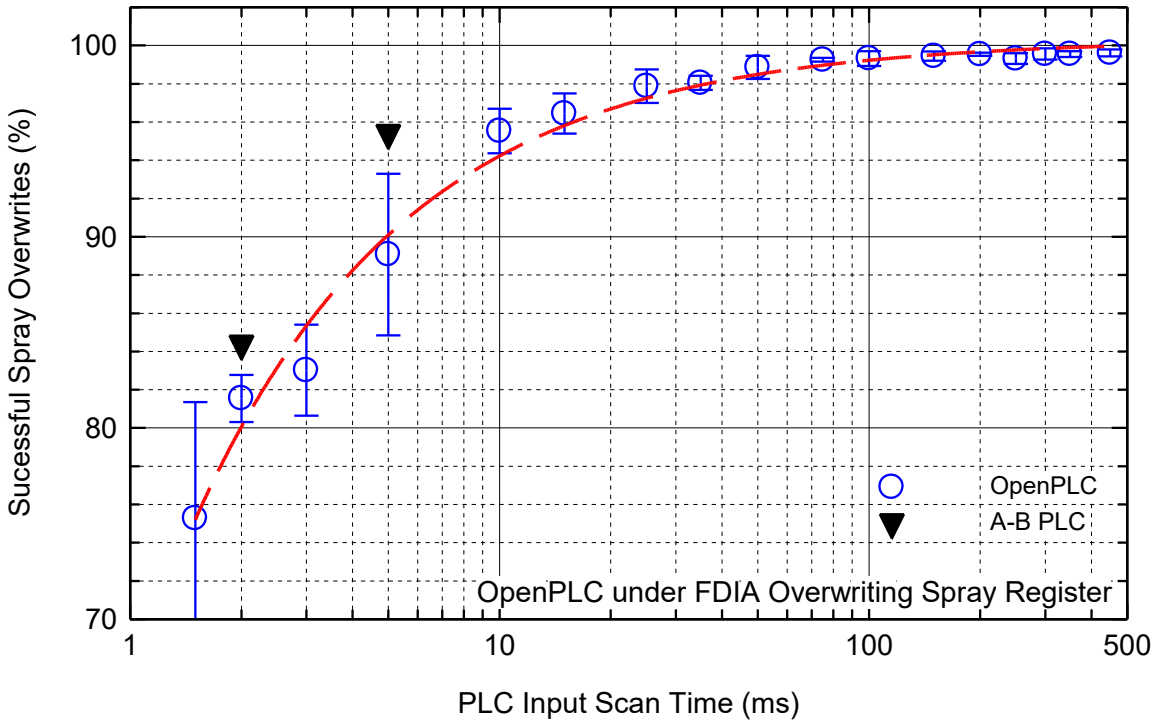

Fig. 3.14. Effect of input scan time on the success percentage to overwrite the Modbus holding register for the emulated PLC during the simulated FDIA targeting water spray control.

3.3.3. Modified OpenPLC Source Code on Overwriting Modbus Holding Registers

Based on the investigation results presented in Section A.1.2 a modified version of the OpenPLC source code is developed to improve the consistency of the actual scan time of the PLC with the input scan time. The modified OpenPLC source code is applied to the emulated pressure PLC to determine the effect of its response to a simulated FDIA. Fig. 3.15 compares the results for the modified and unmodified OpenPLC code for input scan times ranging from 1.2 ms to 50 ms and while an FDIA targeting the holding register of the pressure PLC. The comparisons are repeated three times for each of the input scan time values to improve the statistics of the obtained results. At input scan times greater than ~5 ms the results for the original and modified OpenPLC code are the same (Fig. 3.15). At smaller input scan times the modified Open PLC code increased slightly the success percentage of the overwrites by the applied FDIA for the

modified OpenPLC. For the small input scan times, more consistent scan cycle length of the modified code (Section A1.2.) allows the periodic FDIA to effectively overwrite the holding register. These results support using the developed modified OpenPLC code for the emulated PLCs in LOBO NCS.

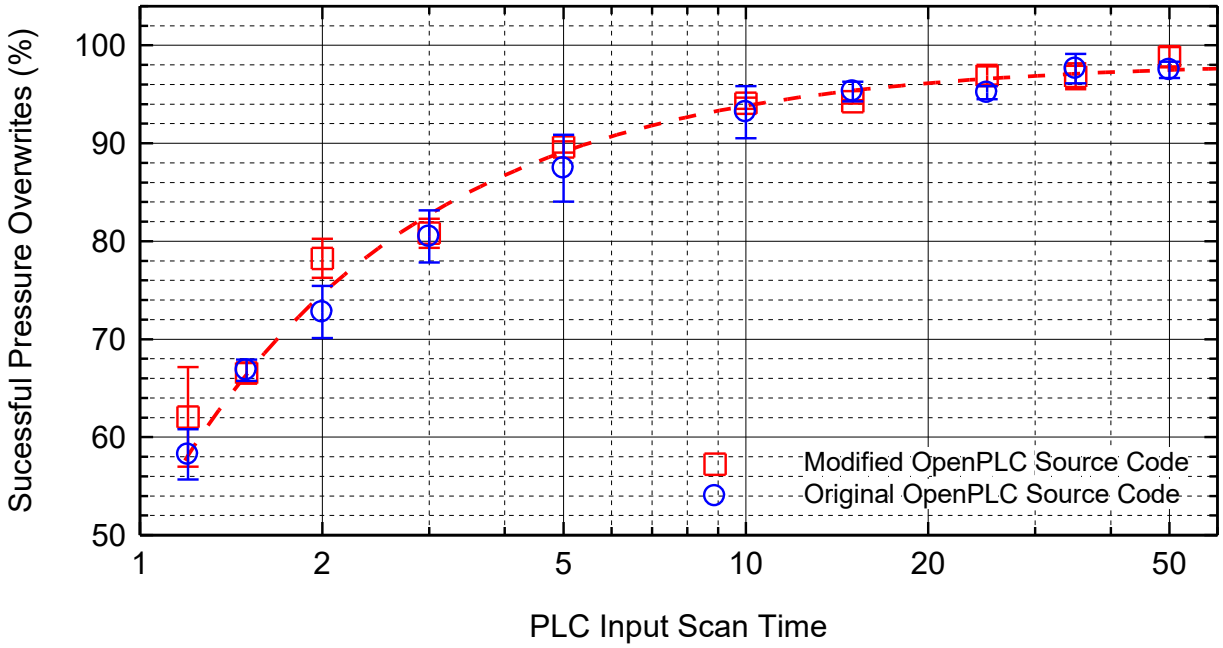

Fig. 3.15. Comparison of the results with original and the modified OpenPLC source codes on the overwrites successes of the emulated pressure PLC' Modbus holding register during FDIA.

3.4. Summary

This section presented the results of investigating the responses of an Allen-Bradley hardware PLC and an emulated PLC while under simulated FDIAs over a range of PLC input scan times. In these simulations, the ManiPIO program within the LOBO NCS platform successfully manipulates the responses of both the Allen-Bradley hardware PLC and the emulated PLC with simulated FDIAs targeting the system pressure and the water spray functions for the pressurizer. The FDIAs either change the input state variables to the Modbus holding registers or overwrite the registers for the output control signals. The first FDIA simulated attempts to overwrite the value of the system pressure to the holding registers of the PLCs and to force them to activate and increase the power to immersed proportional heaters to their peak value and turn on immersed backup heaters.

The second simulated FDIA overwrites the holding register of the water spray control function to disable the water spray into the pressurizer during simulated transient of sequential surge-in and surge-out events. For a PLC scan time of 2 ms, the percentage of the successful overwrites by simulated the FDIA targeting an output holding register is lower for than for the first FDIA overwriting one of the input registers (89.3% vs 99.8% for OpenPLC and 93.6% vs

94.1% for the A-B PLC). This is due to the frequent rate update of the output register by the PLCs' control program. In contrast, the input registers are updated by values communicated by the data broker and communication program each 10 ms simulation timestep.

Results show that the input scan time significantly affects the responses of the emulated and hardware PLCs to the simulated Modbus FDIAs. During normal operation varying the input scan time has a negligible difference in the control the responses of both PLCs are close, regardless of the value of the input scan time. However, when subject to the simulated FDIAs, the percentage of the successful overwrites increases with increased the input scan time. In the first FDIA simulating overwriting an input holding register, when the value of the input scan time approached ~100 ms the ManiPIO program successfully overwrote the pressure value > 98% of the time. In the second simulating FDIA overwriting an output holding register, when the input scan time of the PLC increases beyond~ 50 ms the simulated FDIA successfully supersede the true spray rate value > 99% of the time. This behavior is consistent when using the modified OpenPLC source code. The changes made to the source code do not alter the response of the emulated PLC in the simulated FDIAs.

4. Summary and Conclusions

This report details the work performed to develop and demonstrate the capabilities of the Nuclear Instrumentation & Control Simulation (NICSim) platform developed at the University of New Mexico, in collaboration with Sandia National Laboratory under a DOE NEUP 2018 award. This platform is based on the NICSIM architecture with expanded capabilities to investigate simulated cyberattacks on PLCs within digital I&C systems of a representative PWR plant. However, the platform modularity makes expandable to different nuclear reactor power plant types, including small modular and micro reactors for terrestrial electricity generation. The documented effort successfully links developed Simulink physics-based models of the emulated PLCs in the I&C systems in a representative PWR plant to physics-based models of the different components and the integrated PWR plant.

The LOBO NCS platform investigated the response of the physics-based Simulink models of a representative PWR, and components linked to emulated PLCs during nominal operation transients and when PLCs are targeted by simulated FDIAs. The simulated reactor startup scenario in real time begins from zero power hot critical condition and continues until reaching nominal full reactor power of 3,373 MW_{th}. The emulated PLCs successfully bring the plant to steady state nominal full power at the end of the startup scenario. The individual PLCs function independently but act in concert to maintain values of the reactor's state variables within preprogrammed setpoints. The short communication time between the Simulink model and the PLCs in the LOBO NCS platform enabled the PLCs to ensure smooth control feedback during the simulated startup scenario. Results demonstrate the capabilities of the LOBO NCS platform to couple a multitude of independently running controllers to the Simulink models of a representative nuclear plant and individual plant components.

The LOBO NCS platform also investigated and compared the response of the plant during the same scenario of the nominal startup transient to that of introducing a simulated FDIA targeting the Pressure PLC. The FDIA repeatedly overwrote a false low system pressure to the PLC's holding register. The simulated FDIA initially caused a rapid increase in the system pressure by manipulating the Pressure PLC to increase the power to the submerged proportional heaters and turn on the submerged backup heaters in the pressurizer. The FDIA attempted to overwrite the system pressure every timestep. However, the LOBO NCS communication interface occasionally wrote the true value of the system pressure to the PLC during the FDIA. In response, the PLC sends commands to activate the immersed heaters in the pressurizer when the manipulated pressure value is lower than nominal, and to turn on the water spray into the vapor region of the pressurizer when the true system pressure value is written. These isolated events did not impact the steady rise in the system pressure during the simulated FDIA. The emulated PLCs linked to the Simulink PWR plant model attempted to maintain the plant's operation state variables within their preprogramed setpoints. This limited the impact of the simulated FDIA on calculated state variables for other plant components.

Results demonstrated the inability of the simulated FDIA to overwrite the registers of the PLC 100% of the time. This behavior was further investigated to determine the effects of the

programmed input scan time of the PLCs on their response during simulated cyberattacks. Calculated are the responses of an Allen-Bradley hardware PLC and an emulated PLC using OpenPLC while under simulated FDIAs over a range of PLC input scan times. In these simulations, the FDIAs attempted to manipulate the responses of both the Allen-Bradley hardware PLC and the emulated PLC by either changing the input state variables to the Modbus holding registers for the system pressure or overwriting the registers for the output control signals for the water spray rate in the pressurizer. The first FDIA attempted to overwrite the value of the system pressure to the holding registers of the PLCs to force them to activate and increase the electric power to the immersed proportional heaters to its peak value and turn on the immersed backup heaters. The second simulated FDIA overwrote the holding register of the water spray control function to disable the water spray into the pressurizer during the simulated surge-in and surge-out transient.

The values of the input scan time significantly affect the responses of the emulated and hardware PLCs to the simulated Modbus FDIAs. During normal operation varying the input scan time negligibly affects the responses of both PLCs, regardless of the value of the input scan time. However, when the PLCs are subject to simulated FDIAs, the percentage of the successful overwrites increases with increased the input scan time. For a given scan time, the rate of successful overwrites by the simulated FDIA is consistently higher for the hardware Allen-Bradley PLC than the emulated PLC with OpenPLC. The difference is attributed to the ways the two PLCs manage Modbus TCP communication during the operating scan cycles. The response of the emulated PLC using OpenPLC is consistent with that of a PLC with modified OpenPLC source code for enhancing consistency of the scan times. The performed modifications did not alter the PLC's response to simulated FDIAs, thus support future research using the LOBO NCS platform.

Results show that the configuration settings of a PLC, such as its programmed input scan time, strongly affect how it responds to a potential cyberattack scenario. With short input scan time, the emulated and commercial hardware PLCs are less consistently vulnerable to the Modbus TCP FDIA generated by the ManiPIO program. The responses of the PLCs to different cybersecurity events help identify signs that the inputs of the PLCs are being manipulated and their operation is compromised.

The LOBO NCS platform linking multiple and independently operating emulated or hardware PLCs to a single Matlab Simulink model simulate smooth reliable control during simulated operation transients of a representative PWR plant and enables cybersecurity investigations. Examples are studying the effects of simulated cyberattacks on PLCs in the plant's digital I&C systems on both the behavior of the targeted controllers as well as the responses of the unaffected PLCs as they attempt to state variable on the plant within their preprogrammed setpoints. Future research using the LOBO NCS platform can study the effects of a successful cyber-compromise of the digital I&C system and help identify signs that the system is under cyberattack. This is addition to using the LOBO NCS platform as an I&C testbed for developing the next generation of cybersecurity and autonomous control technology and

methods of terrestrial nuclear reactor power plants and space nuclear power systems, and other energy infrastructure systems.

5. Acknowledgements

The DOE Office of Nuclear Energy's Nuclear Energy University Program funded this research under Contract No. Nu-18-NM-UNM-050101-01 to the University of New Mexico. Any opinions, findings, and conclusions or recommendations expressed in this publication are those of the author(s) and do not necessarily reflect the views of the U.S. Department of Energy.

Sandia National Laboratories is a multi-mission laboratory managed and operated by National Technology and Engineering Solutions of Sandia, LLC, a wholly owned subsidiary of Honeywell International, Inc., for the U.S. DOE's National Nuclear Security Administration under contract DE-NA-0003525. The views expressed in the article do not necessarily represent the views of the U.S. DOE or the United States Government.

Authors thank Sandia National Laboratories for the loan of the Allen-Bradley PLC used in these in the analyses on the LOBO NSC Platform.

6. References

Allen-Bradley, 2013. *1769 CompactLogix Controllers User Manual*, Rockwell Automation Publication 1769-UM011I-EN-P - February 2013.

Alves, T., Morris, T., 2018. "OpenPLC: An IEC 61,113-3 compliant open source industrial controller for cyber security research," *Computers & Security*, **78**, pp. 364-379.

Busquim E. Silva, R.A., et al. 2020. "Development of the Asherah Nuclear Power Plant Simulator for Cyber Security Assessment," *Proc. IAEA International Conference on Nuclear Security 2020*, Vienna, Austria.

Dragos, Inc., 2017a, "TRISIS-Hatman Malware Analysis of Safety Systems Targeted Malware," https://dragos.com/wp-content/uploads/TRISIS-01.pdf.

Dragos, Inc., 2017b. CRASHOVERRIDE, Analysis of the Threat to Electric Grid Operations version 2.20170613, www.DRAGOS.com.

El-Genk, M.S., Schriener, T.M., 2022, "Modeling and Simulation Capabilities for Nuclear Cybersecurity Investigations of a Representative PWR Plant and a Space Reactor Power System," in Nuclear Power Plants: Recent Progress and Future Directions, J.K. Compton (Ed.), Nova Science Publishers, Hauppauge, NY, 2022.

El-Genk, M.S, Schriener, T.M., Lamb, C., Fasano, R., Hahn, A., 2019, Implementation and Validation of PLC Emulation and Data Transfer, Report No. UNM-ISNPS-02-2019, Institute for Space and Nuclear Power Studies, University of New Mexico, Albuquerque, NM, USA

El-Genk, M.S, Schriener, T.M., Hahn, A., Altamimi, R., Fasano, R., Lamb, C., 2020a, Emulated Programmable Logic Controllers for the Protection and Safety Monitoring and Operation I&C Systems in a Representative PWR Plant for Cybersecurity Applications, Report No. UNM-ISNPS-03-2020, Institute for Space and Nuclear Power Studies, The University of New Mexico, Albuquerque, NM, USA.

El-Genk, M.S, Schriener, T.M., Altamimi, R., Hahn, A., 2020b, A Physics based, Dynamic Model of a Pressurized Water Reactor Plant with Programmable Logic Controllers for Cybersecurity Applications, Report No. UNM-ISNPS-02-2020, Institute for Space and Nuclear Power Studies, The University of New Mexico, Albuquerque, NM, USA.

El-Genk, M.S, Schriener, T.M., Hahn, A., Fasano, R., Lamb, C., 2021, Validation of LOBO Nuclear CyberSecurity (LOBO NCS) Platform and Demonstration of Manipulate Process I/O (ManiPIO) Framework for Cybersecurity Testing and Evaluation, Report No. UNM-ISNPS-01-2021, Institute for Space and Nuclear Power Studies, The University of New Mexico, Albuquerque, NM, USA.

El-Genk, M.S., Schriener, T.M., 2022, "A Cybersecurity Platform for Simulating Transient Responses of Emulated Programmable Logic Controllers in Instrumentation and Control Systems for a PWR Plant," Journal of Cyber Security Technology, 6(1-2), pp. 66-90.

El-Genk, M.S., Altamimi, R., Schriener, T.M., 2021, "Pressurizer Dynamic Model and Emulated Programmable Logic Controllers for Nuclear Power Plants Cybersecurity Investigations," Annals of Nuclear Energy, 154, 108121.

Hahn, A., Schriener, T.M., El-Genk, M.S., 2020b. "Selection and validation of fast and synchronous interface to the controller of a space nuclear reactor power system," Proceedings of the 2020 28th Conference on Nuclear Engineering Joint with the ASME 2020 Power Conference ICONE28-POWER2020, August 2-6, 2020, Anaheim, California, USA, paper ICONE28-POWER2020-16237

International Association for the Properties of Water and Steam, 2007, The International Association for the Properties of Water and Steam Revised Release on the IAPWS Industrial

Formulation 1997 for the Thermodynamic Properties of Water and Steam, Lucerne, Switzerland, IAPWS R7-97(2012)

Karnouskos, S., 2011. "Stuxnet Worm Impact on Industrial Cyber-Physical System Security," in proceedings IECON 2011 - 37th Annual Conference of the IEEE Industrial Electronics Society, Melbourne, VIC, Australia, 7-10 November 2011, DOI: 10.1109/IECON.2011.6120048.

Metzger, J.D., El-Genk, M.S., Parlos, A.G., 1991. "Model-Reference Adaptive Control with Selective State-Variable Weighting Applied to a Space Nuclear Power System," *J. Nuclear Science and Engineering*, **109**, pp. 171-187.

National Research Council, 1997. Digital Instrumentation and Control Systems in Nuclear Power Plants, Safety and Reliability Issues, Final Rep., National Academy Press, Washington, D.C.

Nuclear Energy Institute, 2010, "Cyber Security Plan of Nuclear Power Reactors," NEI Technical Report NEI 08-09 [Rev.6].

MathWorks, 2020, "Matlab & Simulink R2020a," Natick, Massachusetts, United States, https://www.math works.com/

Schriener, T., El-Genk, M.S., 2022. "Response of Programmable Logic Controllers of the Pressurizer in a Representative PWR Plant Following a False Data Injection," in proceedings 12th Nuclear Plant Instrumentation, Control and Human-Machine Interface Technologies (NPIC&HMIT 2021), paper No. 34539, virtual meeting, June 14-17, 2021.

Trask, D., Jung, C., MacDonald, M., 2014. "Cybersecurity for Remote Monitoring and Control of Small Reactors," *Proc. 19th Pacific Basin Nuclear Conference (PBNC 2014)*, Vancouver, British Columbia, Canada, 24-28 Aug. 2014.

VMware, 2019. VMware Workstation 15 Pro.

Zhang, F., Coble, J.B., 2020. "Robust localized cyber-attack detection for key equipment in nuclear power plants," *Progress in Nuclear Energy*; **128**; 103446.

A. Characterization of OpenPLC in LOBO NCS

The developed emulated PLCs of the I&C architecture of the representative PWR plant in the LOBO NCS platform use an open-source, non-proprietary OpenPLC framework. PLC vendors use their own proprietary operating systems and control software, and frequently employ vendor-proprietary ICS communication protocols. Thus, they are less suited for general research cybersecurity testing and training platforms (Alves and Morris 2018). The OpenPLC software developed by Alves and Morris (2018) presents an open-source option for developing PLCs capable of running control programs written in IEC 61131-3 standard PLC programming languages.

The OpenPLC runtime installed in Linux and Windows platforms supports communication using both Modbus and DNP3 ICS protocols. OpenPLC also supports connections to physical sensors and actuators by installing the software on minicomputers like the Raspberry Pi with analog I/O connections. In this arrangement, the Modbus register values can be linked to the analog inputs and outputs on the motherboard or connected I/O module. This makes OpenPLC a useful choice for cybersecurity platforms for education at academic institutions and professional training in general (Alves and Morris 2018).

The open-source OpenPLC program selected to emulate the PLCs in the Lobo NCS Platform supports IEC61131-3 standard PLC programming languages and the communication using the commonly used Modbus TCP and DNP3 over TCP ICS protocols (Alves and Morris 2018). It also supports popular SCADA protocols for integration with supervisory control systems and has an easy-to-use graphical interface for monitoring the PLC's operation. This open-source and nonproprietary program allows unrestricted academic research without concerns of compromising sensitive information, unlike the PLCs used in actual industrial facilities or commercial nuclear plants. Commercial PLCs typically have a consistent scan time enforced by a real time clock.

This consistency may not necessarily be the case for the software based OpenPLC runtime. Therefore, a series of studies are performed to characterize the behavior and quantify the performance of OpenPLC when integrated into the LOBO NCS platform. The first investigates the actual scan time for the OpenPLC runtime and how well it emulates the performance of hardware PLCs. The second study investigates the Modbus TCP communication between the OpenPLC and the data communication interface in the LOBO NCS platform. This study quantifies the effects of the simulation timestep, the PLC scan time, and the communication network on the timing and reliability of the Modbus TCP communication involving sending signals from a Matlab Simulink model to an OpenPLC running a Raspberry Pi minicomputer.

A.1. Characterization of OpenPLC Scan Time

PLCs perform their control operations in a repetitive cycle known as the scan cycle. During each cycle, the PLC performs three specific operations (Figure 1.2), namely: (a) reading (or scanning) the input values, (b) executing the control logic program using the input values, and (c) writing the output values for subsequent control response (Alves and Morris 2018). The

minimum time required to complete a scan cycle depends on number of factors including the speed of the processor, the complexity of the control logic program, and the number of inputs and outputs which must be read and written to. It is essential that the PLC completes these operations within the desired scan time to send the control signals at the expected frequency without delays. The performed study characterized the OpenPLC scan time and examined the consistency of complying with the scan time specified by the user. Also investigated are options for altering the OpenPLC source code to improve the consistency of the scan times while running the PLC.

A.1.1. Methodology

The PLCs' scan cycle performed within a program loop within the OpenPLC source code with subroutines for its distinct functions (Table A.1). These functions performed sequentially within the loop. The GlueVars subroutine associates the variables from the PLC's uploaded control program to the OpenPLC memory buffer pointers. These pointers specify the memory buffer area where OpenPLC temporarily stores values as needed. The updateBuffersIn and updateBuffersOut subroutines update the internal buffers with the current values of physical input and output pins in the I/O module of the PLC. Both the Pthread_mutex_lock (&BufferLock) and the Pthread_mutex_unlock (&BufferLock) subroutines lock and unlock the process threads accessing the buffers to avoid simultaneous read and write attempts to the same memory location.

The updateCustomIn and updateCustomOut subroutines provide the user a location in the OpenPLC program to insert their own custom input and output functions into OpenPLC. The updateBuffersIn_MB and updateBuffersOut_MB functions update the internal memory buffers based on the current received values by Modbus communication. The Config_run subroutine executes the PLC logic program. Lastly the sleep_until subroutine forces the loop to sleep at the end of the cycle to match the cycle time of the PLC and the input scan time.

The UpdateTime subroutine compares the time during the current scan cycle to that expected had the PLC functioned exactly at the input scan time. The OpenPLC program adjusts the sleep period based on not just the previous scan cycle time, but of the average over the course of the PLC's operation period. The program decreases the sleep time when the PLC average time is longer than the input scan time until the program catches up to the target average scan time. Conversely, when the PLC average scan time runs ahead of the input scan time the sleep time increases to slow down the PLC until average scan time matches.

To investigate the actual scan time of OpenPLC, the source code is modified to record the scan time each cycle. The recorded values are stored in memory and written to an output file for analysis when the PLC program is commanded to shut down. The recorded actual scan time within the OpenPLC program's main loop equals the sum of the time required for each of its subroutines (Table A.1) plus the sleep time. The tests of determining the scan time during operation are performed using the modified version of OpenPLC running on both a Raspberry PI 3B minicomputer and a VM running the Raspian OS. In each test the OpenPLC program runs 5000 scan cycles to ensure sufficient sample size for comparison.

Table A.1. OpenPLC main loop subroutines and functions in order of execution

Subroutine	Function
GlueVars	Glues variables from the IEC program to the OpenPLC memory buffer pointers.
updateBuffersIn	Updates the internal buffers to reflect the actual state of the input pins.
Pthread_mutex_un/lock (& Buffer lock)	Locks/unlocks the mutex to protect access to the buffers in a threaded environment.
updateCustomIn	Updates the internal input buffers with the values indicated by the user.
updateBuffersIn_MB	Updates internal buffers for functions using Modbus to reflect the actual input state.
handelSpecialFunctions	Used for any special function to be used in the program.
Config_run	Executes the PLC logic program.
updateCustomOut	Updates the internal output buffers with the indicated values by the user.
updateBuffersOut_MB	Updates internal buffers for functions using Modbus to reflect the actual output state.
UpdateBuffersOut	Updates the internal buffers to reflect the actual state of the output pins.
UpdateTime	Updates the time after completing the calculation and determines the sleep time to match the input scan time.
Sleep_until	Makes the program sleeps after the calculations are done and to match the input scan time.

Table A.2. Recorded scan time of the Raspberry Pi OpenPLC for different input scan times.

Input scan time (ms)	Mean recorded scan time (ms)	Max. recorded scan time (ms)	Min. recorded scan time (ms)
0.01	0.078020991	0.077864	0.064947
0.8	0.801547552	127.749688	0.063646
1.0	0.998856296	147.652865	0.068125
2.0	1.998566	133.468958	0.075417
3.33	3.29829616	126.849844	0.075938
4.0	3.99814	168.326198	0.075312
5	4.998036719	121.332344	0.075052001
6.67	6.664421491	161.451927	0.075989999
7	6.997799738	160.632761	0.076562999
8	7.997850953	123.287031	0.075468999
10	9.997705069	129.073177	0.077135
20	19.99686789	124.000885	0.079115
23	22.9959836	162.601979	0.077864
26.67	26.66245814	158.008802	0.079791

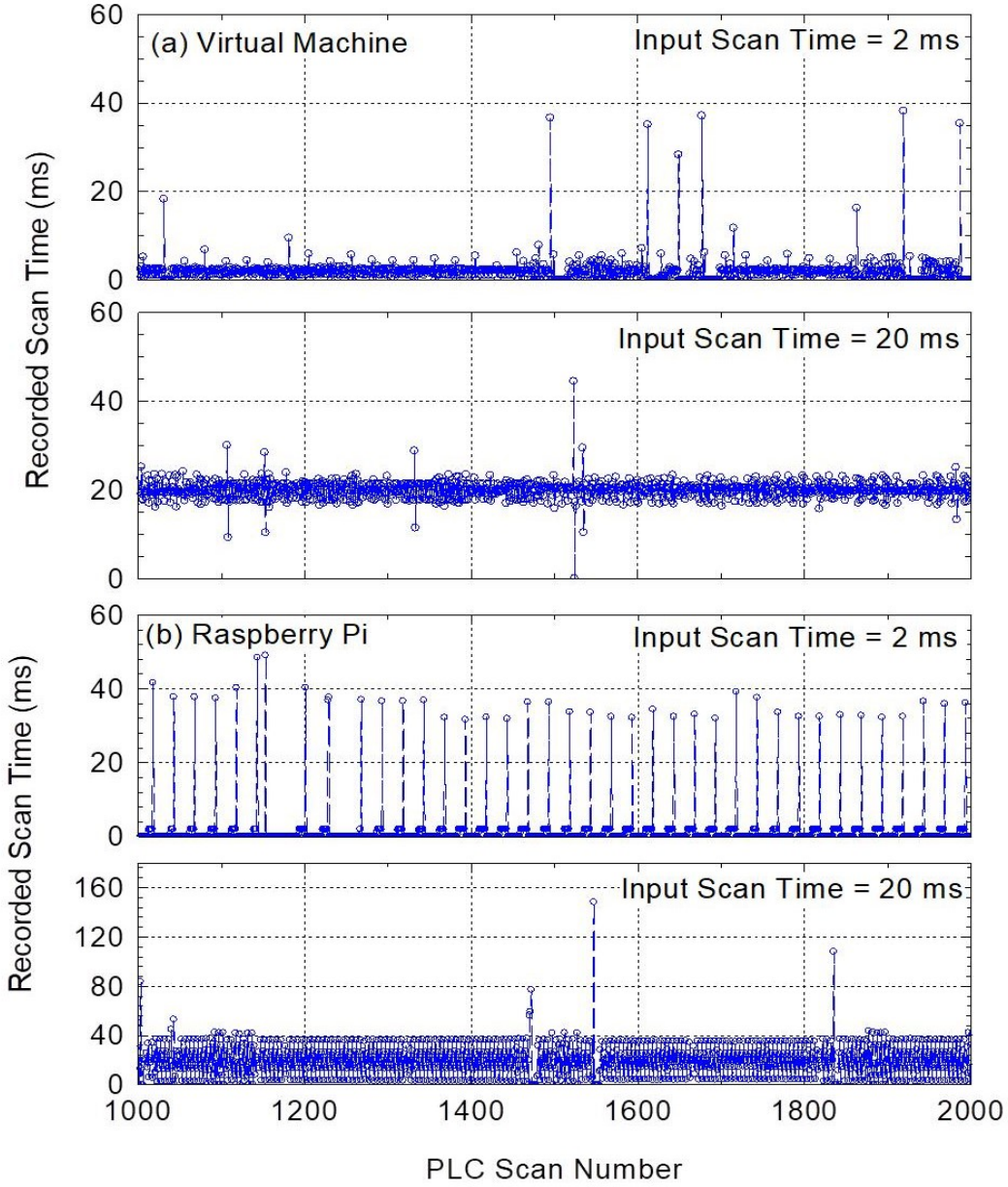

Fig. A.1. Recorded scan time for OpenPLC running on: (a) virtual machine with Raspian OS, and (b) Raspberry Pi minicomputer.

A.1.2. Scan Time Characterization Results

The characterization effort records and compares the scan times of OpenPLC running on the Raspberry Pi and virtual machine. Table A.2 lists the recorded scan times of the OpenPLC program running on the Raspberry Pi for input scan times ranging from 0.01 to 26.67 ms. Listed in Table A.2 are the recorded mean, minimum and maximum scan times for each input scan time case. For all the cases run on the Raspberry Pi the smallest recorded scan time is approximately 63.6 µs, with a minimum ranging up to 79.8 µs. The longest scan cycle recorded for each input scan cycle varies significantly and is much longer than the input scan time. The OpenPLC sleep time function kept the mean scan time equal to the input scan time except for the case with an

input scan time of 0.01 ms. The results listed in Table A.2 clearly show that the OpenPLC program on the Raspberry Pi is incapable of running in synch at this short input scan time. The input scan time tests are repeated for the OpenPLC runtime on the VM, and the results obtained using the Raspberry Pi are compared to determine the effect of the platform on the recorded scan time (Figure A.1). This figure compares the recorded scan times for OpenPLC running on the VM and Raspberry Pi for input scan times of 2 and 20 ms. In both cases, the recoded scan time is not constant, with a series of spikes of much longer or shorter recorded scan times than those specified in the input. Note that the cases run on the VM resulted in fewer spikes than those on the Raspberry Pi, suggesting that some OpenPLC subroutines take more time on the Raspberry Pi.

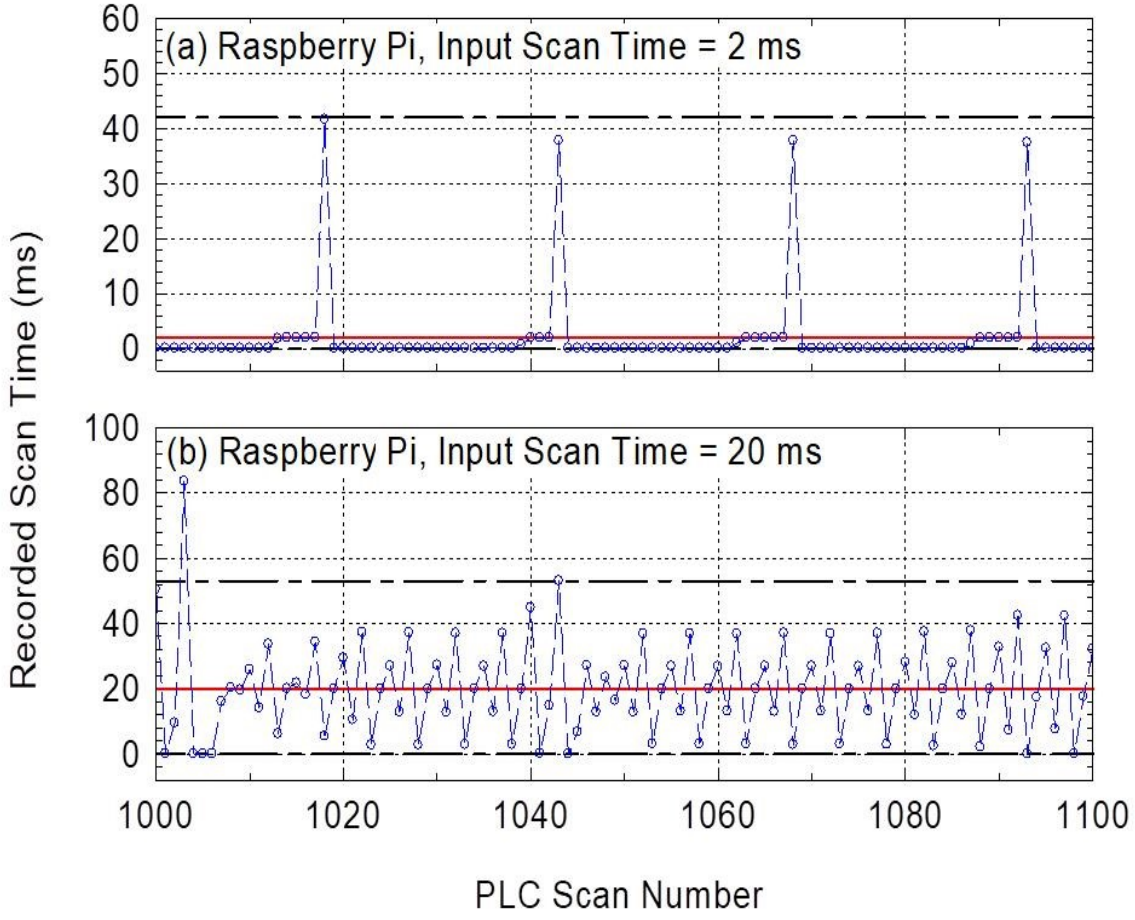

Fig. A.2. Recorded scan times for OpenPLC on Raspberry Pi.

Figure A.2 shows the recorded scan times for the Raspberry Pi in these scan cycle with spikes for the 2 ms and 20 ms input scan times. Periodically, OpenPLC's scan cycle is much longer than the input scan time, resulting in spikes in the recorded scan time. Subsequently, the sleep time function in OpenPLC reduces the sleep time near zero to bring the mean scan time into alignment with the input value of the scan time. During these catch-up cycles, the recorded OpenPLC scan time is well below the input scan time. Once the PLC mean scan time matches the input time, the recorded scan times become in good agreement with the input scan time until

the next spike event (Fig. A.2a).

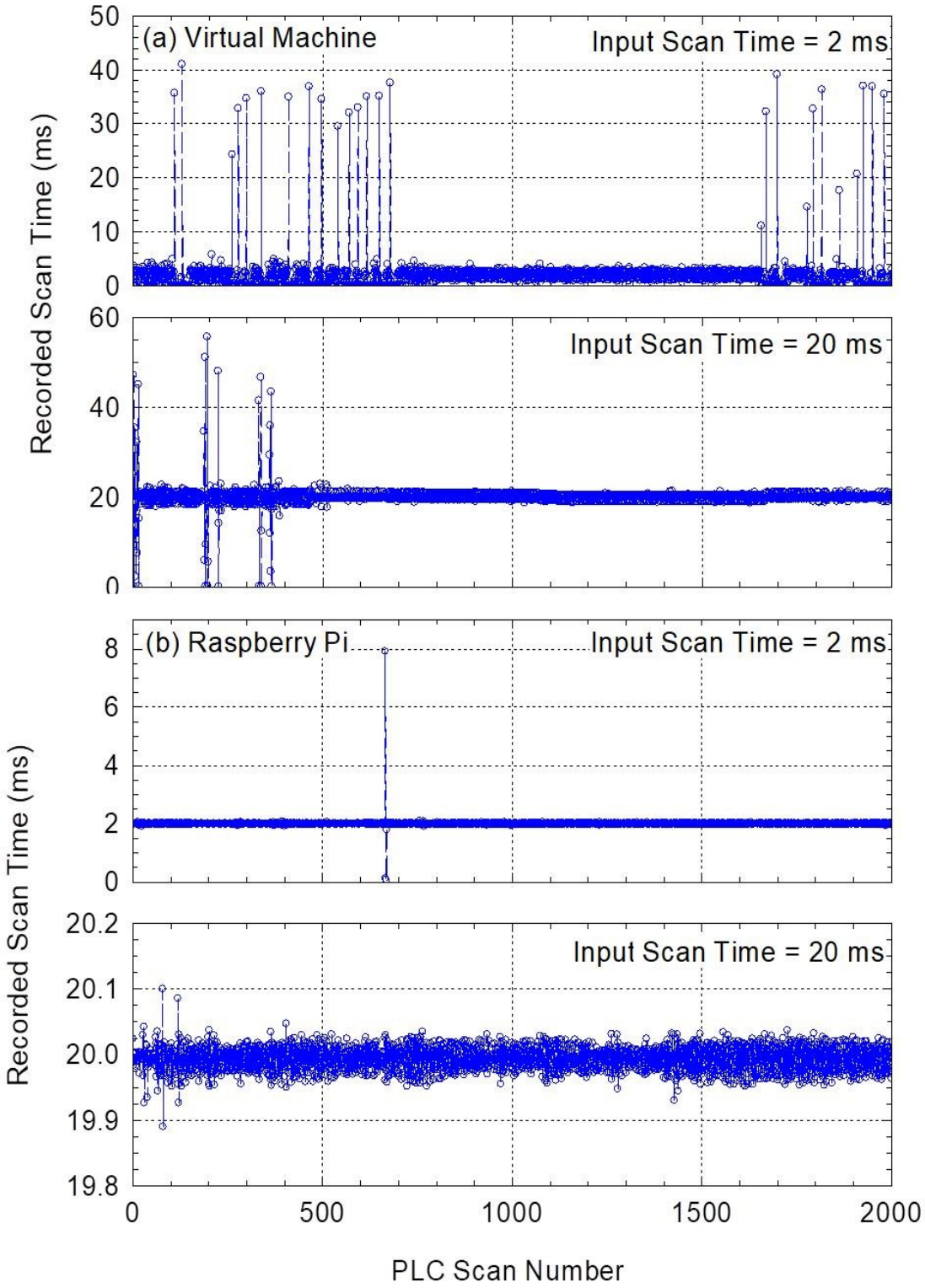

Fig. A.3. Recorded scan times for modified OpenPLC using: (a) virtual machine, and (b) Raspberry Pi.

The examined OpenPLC source code is help determine the cause of the long scan time events during the operation of the PLC. The recorded computation time for each of the subroutines in Table A.1 help identify which function or functions were causing the intermittent delays. This data suggest that the long scan time events are caused by the updateBuffersIn and Pthread_mutex_lock (&Buffer lock) subroutines.

These subroutines are responsible for updating the internal buffers for the values stored in the input pins associated with the memory registers in OpenPLC and for locking the thread during these operations. For integration within the LOBO NCS Platform the memory registers for the physical input pins in the OpenPLC program are not used because the communication to the PLC's holding registers used Modbus TCP instead. Therefore, the updateBuffersIn and updateBuffersOut subroutines and the associated mutex thread lock calls in the OpenPLC source code are disabled. Fig. A.3 presents the recorded scan times for the modified OpenPLC code with these subroutines disabled, running on the VM and Raspberry Pi for the same input scan times of 2 ms and 20 ms input as the cases presented in Figs. A.1-A.2.

The recorded scan time for the Raspberry Pi with the revised source code is much more consistent with the specified input scan time. For the two cases in Fig. A.3 the recorded scan times agree with the input scan times to within ± 0.1 ms, except for a single 8 ms recorded scan time for the case with an input scan time of the 2 ms. The cases presented in Fig. A.3 using the modified version of OpenPLC on the VM show more frequent long scan time events than the Raspberry Pi. However, the frequency decreased faster compared to that obtained using the original OpenPLC source, without modification.

Further testing shows that the remaining spikes in the scan times are due to the handelSpecialFunctions subroutine in OpenPLC. This subroutine could not be disabled without affecting the PLC, as this subroutine could potentially be used for some OpenPLC programs of the emulated PLCs. In summary, while OpenPLC closely matches the mean scan time to the input value over the length of its operation, it displays considerable inconsistency of results among individual scan cycles. This inconsistency was primarily due to two subroutines in the Open PLC main program loop for the scan cycle. The scan time results using the modified OpenPLC source code show improved consistency with the values of the input scan time. This consistency suggests that the modified OpenPLC more closely emulates commercial PLCs which maintain highly consistent timing behavior due to their internal real-time clocks.

A.2. Communication Characterization of OpenPLC on LOBO NCS Platform

The next characterization effort focuses on the communication between OpenPLC and the LOBO NCS interface (Fig. 1.1). The OpenPLC runtime has been used in cybersecurity research and successfully linked to simulation models (Alves and Morris 2018). However, to the best of authors' knowledge, no prior working on characterizing the communication between OpenPLC and a physical system has been reported. The latter is the focus of the work reported in this section of characterizing the communication between OpenPLC and Simulink model using a Python data transfer interface (Fig. A.4).

The performed work investigates the effects of different variables on the communication's cycle time and the reliability of the Modbus TCP communication between the Simulink model and OpenPLC. The cycle time is that elapses between two subsequent input signals sent by the Simulink model to OpenPLC. The communication reliability is defined the correct sequential returned signals received from the PLC as a fraction of the total signals sent by the Simulink model. High reliability of the emulated PLCs is desirable as missing or delaying control signals can alter the transient response of the physics-based model. The obtained characterization results are used to identify the appropriate settings options for the Simulink model and OpenPLC runtime to achieve reliable communication between the PLC and the Simulink data provider on the LOBO NCS test network.

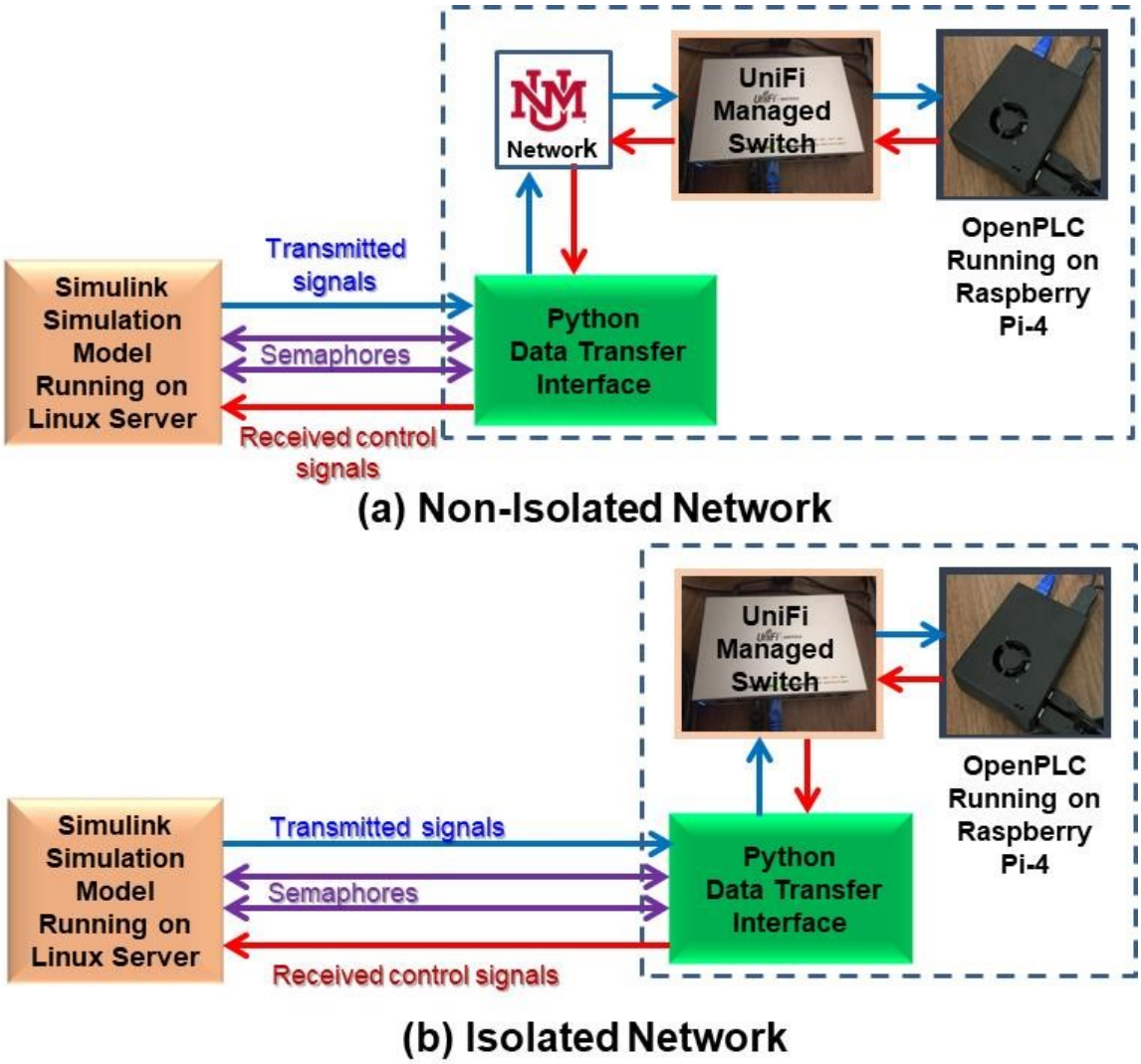

Fig. A.4. A layout of the testing setup for the communication characterization of OpenPLC using isolated and non-isolated networks: (a) Non-Isolated Network, (b) Isolated Network.

A.2.1. Methodology

The LOBO NCS platform is configured with a Simulink model that generates repeating

signals as a data provider and a single hardware PLC. These signals pass to a Python data transfer interface which communicates them to the PLC. The PLC returns the received signals back to the Simulink model through the data transfer interface. The PLC in the performed experiment is a Raspberry Pi 4 minicomputer with the OpenPLC runtime (Fig. A.4). This effort investigates the effects of four different variables on the communication characteristics between the Simulink model and OpenPLC using the Python data transfer interface. These variables are: (i) the type of network setup, (ii) the form of the Simulink generated signals, (iii) the input scan time for the OpenPLC runtime, and (iv) the simulation timestep used in the Simulink model. The investigated effects are for isolated and non-isolated Ethernet networks to link the PLC to data transfer interface (Fig. A.4).

With the non-isolated network, the Modbus packets were communicated through the University of New Mexico's Ethernet network (Fig. A.4a). This setup has the advantage of using high-end networking hardware and the easy access to university online services. However, this is at the expense of increasing the round-trip communication distance and potentially picking up outside traffic noise. The latter avoided using the isolated network setup that routes all the data packets through a local managed ethernet switch. This allows for a more controlled environment with simpler network data capture and shorter routing distances, but at the cost of using modest switching hardware.

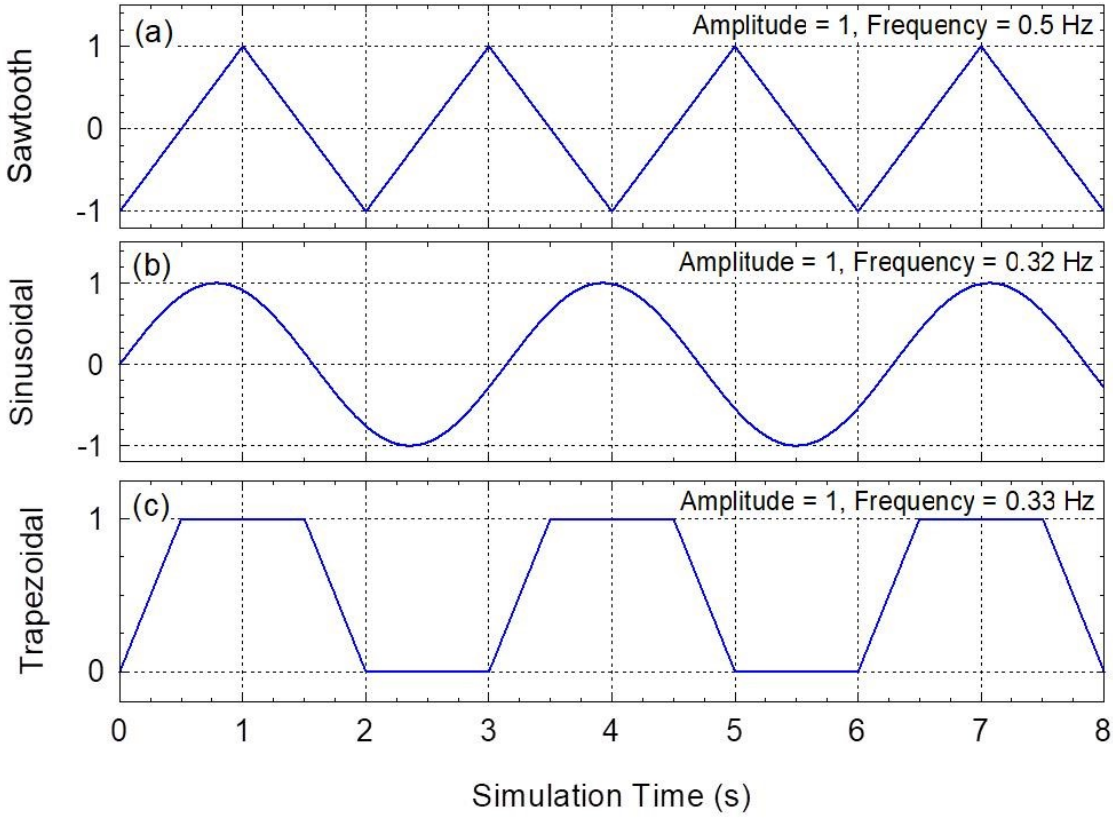

Fig. A.5. Sawtooth, sinusoidal, and trapezoidal repeating signals generated by the Simulink model and communicated to the PLC.

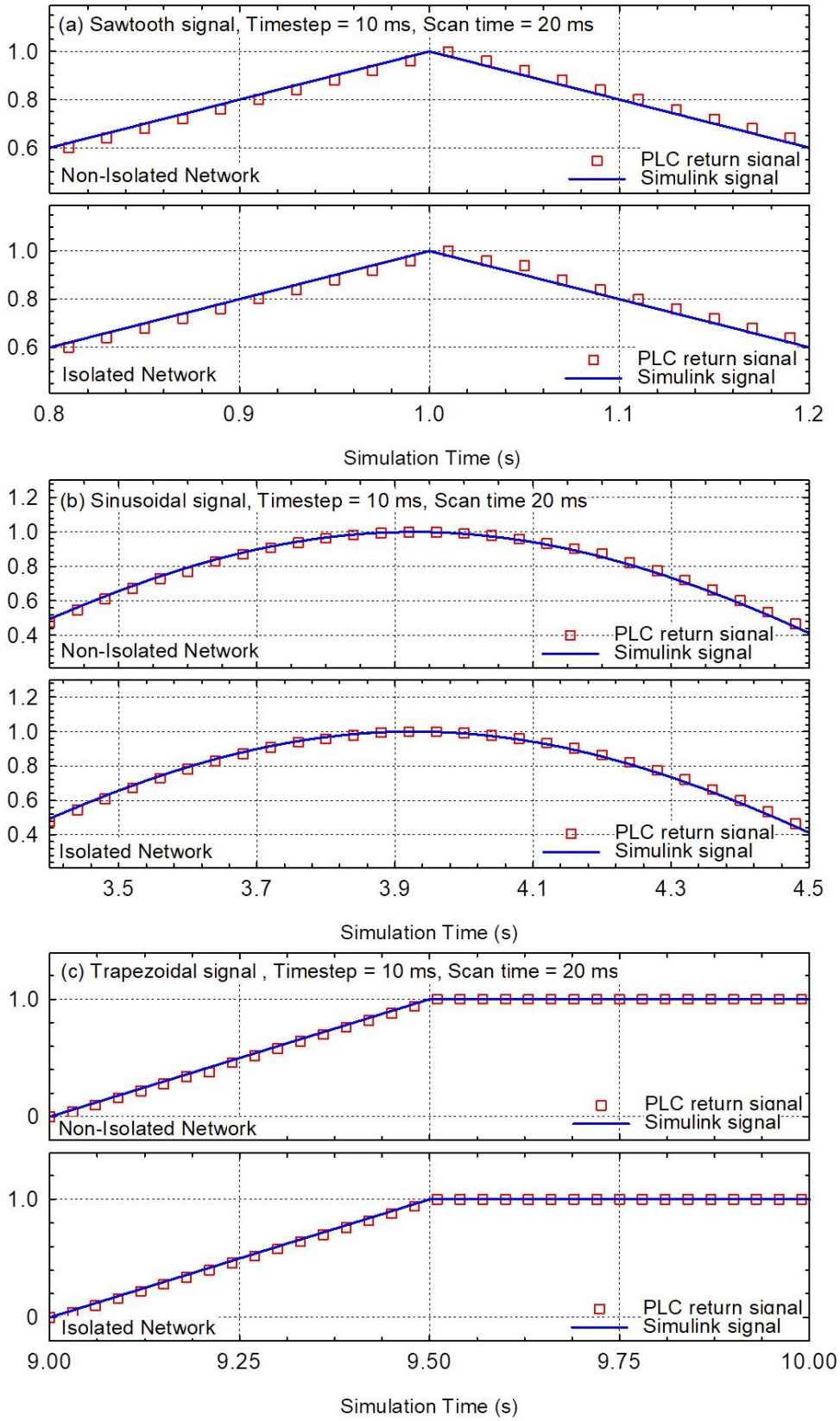

Fig. A.6. Generated and returned signals for 10 ms simulation and 20 ms PLC scan times.

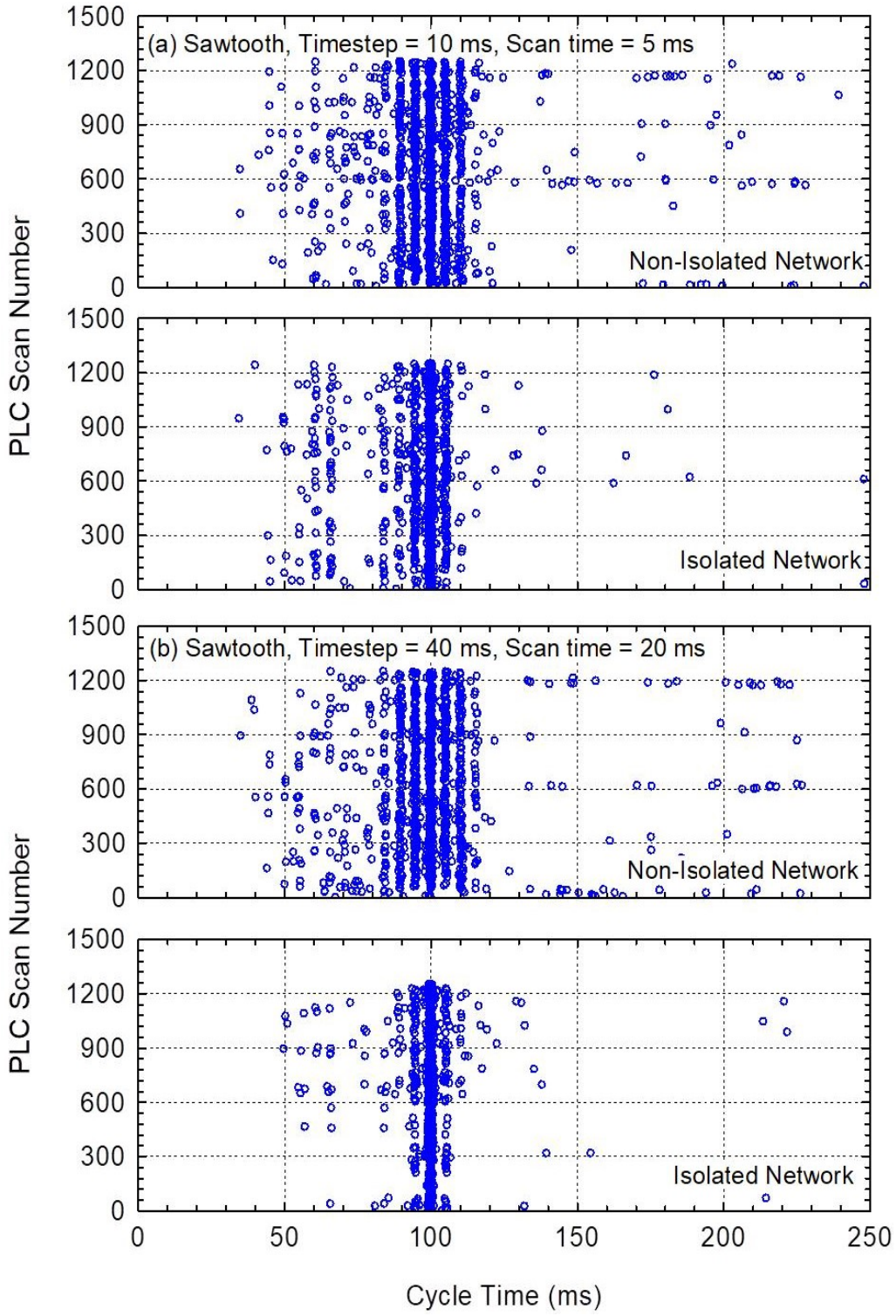

Fig. A.8. Recorded communication of a sawtooth signal on isolated and non-isolated networks.

The performed study also investigated the effect of the type of signal generated and sent by the Simulink model to the PLC on the cycle time and the communication reliability. The three different signal shapes examined are: a sawtooth signal with sharp transitions, a smooth

sinusoidal wave, and a trapezoidal signal (Fig. A.5). The sawtooth signal has an amplitude of 1.0 and a frequency of 0.5 Hz, the sinusoidal and the trapezoidal signals have same amplitude but frequencies of 0.32 Hz, and 0.33 Hz.

The performed study also investigated the effect of the input scan time on the communications reliability between the OpenPLC runtime and the Simulink model. As discussed in Section A.1, the scan time is the duration of time taken by the PLC to go through its operating cycle. This includes the scanning and processing of the input signal through the logic program and writing the value of the output signal to the PLC holding registers. The investigated values of the input scan times varied from 0.83 to 26.67 ms. The last parameter investigated is the effect of the simulation timestep inside the Simulink model on the communication reliability. This investigation used simulation timesteps of 10, 20, and 40 ms to evaluate the response of OpenPLC to the frequency of the signals generated within the Simulink model.

After determining the most probable cycle time for the communication between the Simulink model and the PLC, the communication reliability calculated as a function of the OpenPLC scan time and the most probable cycle time. The communication reliability quantified for scan times ranging from 0.83 to 26.67 ms. Each case generated 1250 signal data points that are sent by Simulink to the OpenPLC and then returned by the PLC to the Simulink model. Each case is repeated 15 times to generate a large database to help accurately quantify the communication reliability at decreased data uncertainty.

A.2.2. Communication Characterization Results

This section presents the results of the communication characterization between OpenPLC and the Simulink simulation using a Python data transfer interface on the LOBO NCS platform. Cases are run for each of the three generated signal types in Fig. A.5 with the Simulink simulation time steps of 10, 20, and 40 ms and OpenPLC input scan times ranging from 0.83 ms to 26.67 ms, for both isolated and non-isolated networks (Fig. A.4). Fig. A.6 compares the sawtooth, sinusoidal, and trapezoidal signals sent by Simulink to those returned by the OpenPLC runtime using both isolated and non-isolated networks. Results in Fig. A.7 are for Simulink timesteps of 10 ms and OpenPLC scan time of 20 ms.

Figures A.7a and A.7b show the recorded data for the communication cycle time when using isolated and non-isolated networks. Fig. A.7a presents the obtained results for 10 ms simulation timestep and 5 ms input scan time, while Fig. A.7b presents the result for 40 ms simulation timestep and 20 ms input scan time. The results show a most probable communication cycle time of ~100 ms for all cases investigated. The data when using the non-isolated communication network shows greater scattering in the recorded cycle times than the data using isolated communication network. The controlled environment of the isolated network results in a more consistent communication times and reduces the digital signal noise, and hence the variance in the recorded data.

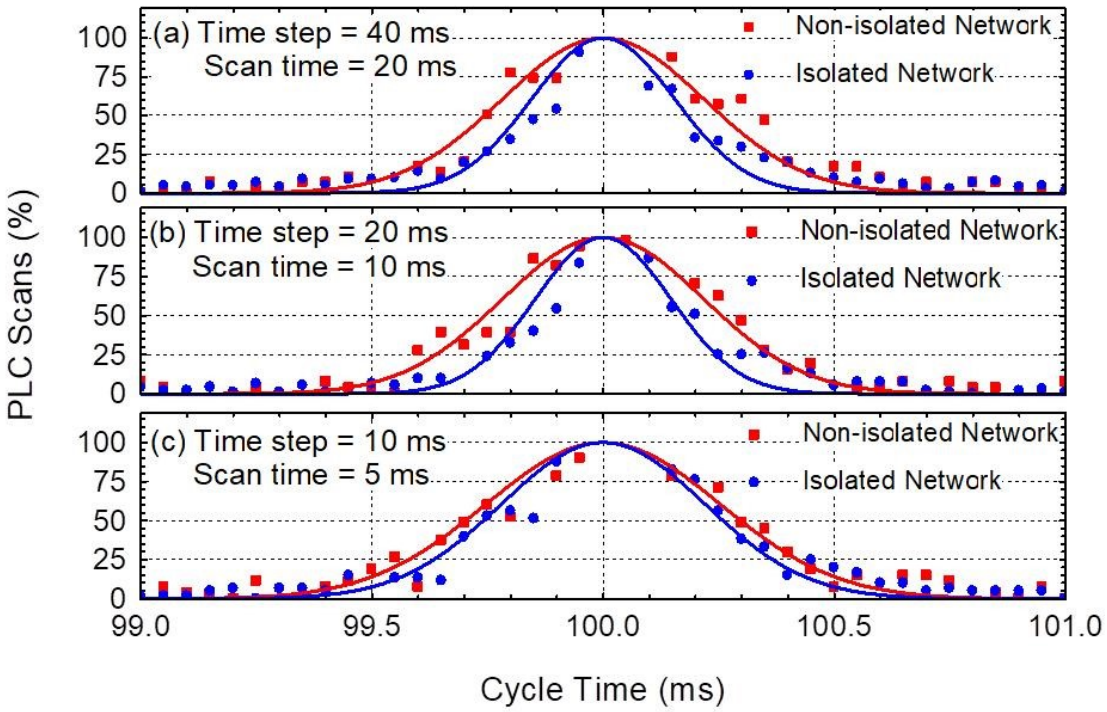

Fig. A.8. Normal distribution of PLC scans versus communication cycle time at different simulation timesteps and scan times.

As demonstrated in Fig. A.8, the recorded cycle time data fits a normal distribution for the cases using either isolated or non-isolated communication networks. The most probable cycle time for all cases investigated is 100 ms, but the deviation in the recorded data is larger with the non-isolated communication network due to the higher traffic noise. The results in Fig. A.8 are for the sawtooth signal, which are consistent with those obtained for the sinusoidal and trapezoidal signals (Fig. A.5). The recorded data analyzed to quantify the communication reliability as a function of the PLC scan time and the determined most probable cycle time.

Fig. A.9 compares the communication reliabilities for the sawtooth, sinusoidal, and trapezoidal signals with Simulink simulation timesteps of 10, 20, and 40 ms, and using isolated and non-isolated communication networks. The reliability values plotted versus the normalized PLC process time, τ, which is defined as the PLC's input scan time divided by the determined most probable cycle time of 100 ms (Fig. A.7-A.8). Each point represents the mean value of τ based on the results of the 15 repeated test runs. For all three signal types investigated the communication reliability decreases as the normalized PLC process time, τ, increases (Fig. A.9). The average reliability values for the three signal types investigated are similar for the different values of the simulation timestep used. The generated data for the sawtooth and sinusoidal signals agree with the average trend to within ±0.3% (Figs. A.9a and b), and to within ±0.2% of that for the trapezoidal signals. For the sawtooth and sinusoidal signals, the results indicate that τ needs to be < 0.06 to achieve almost 100% communication reliability. The results using both isolated and non-isolated networks are the same (Figs. A.9a-b).

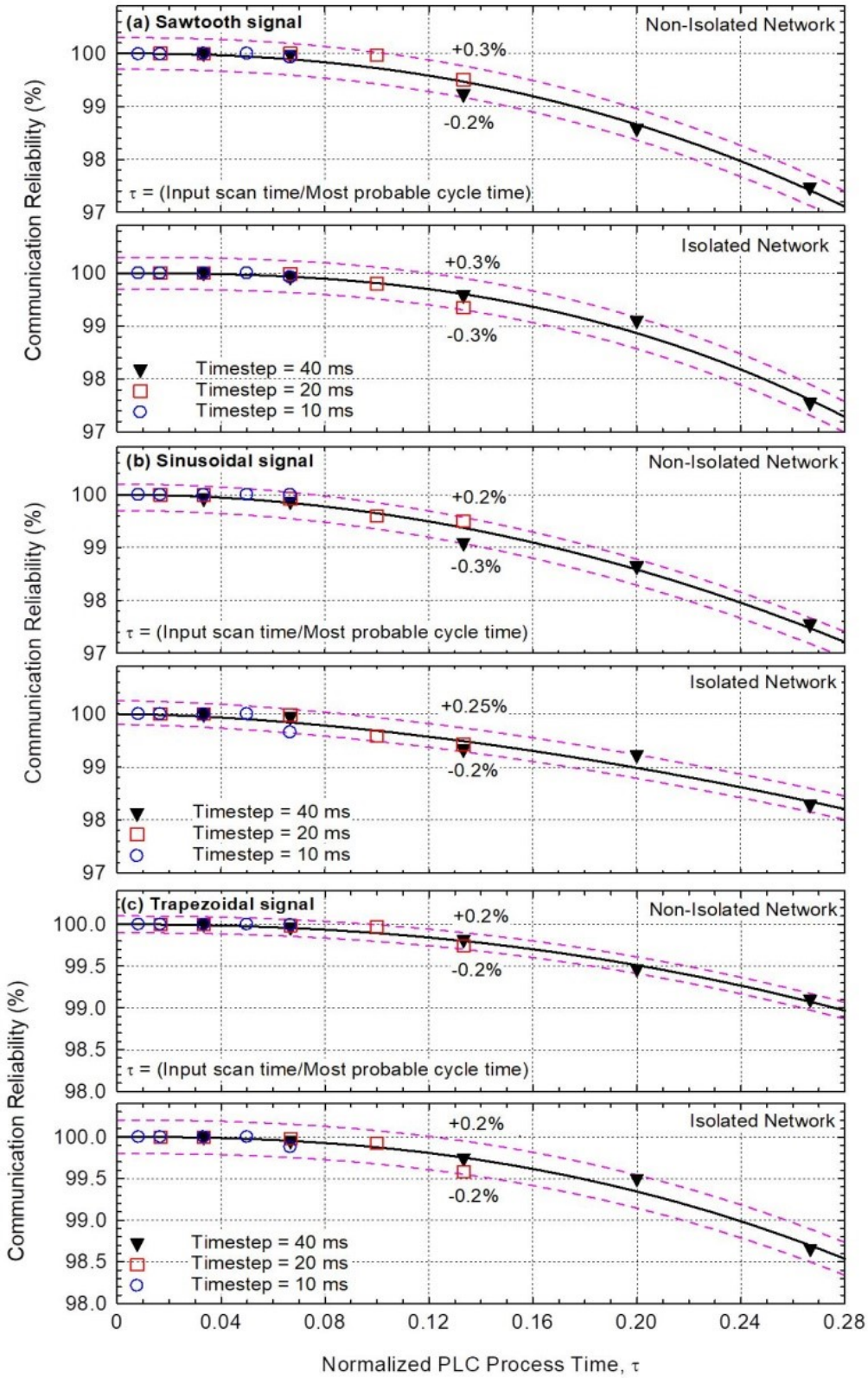

Fig. A.9. Communication reliability of Simulink model with OpenPLC using a Python data transfer interface for different signal types: (a) Sawtooth, (b) Sinusoidal, and (c) Trapezoidal

The communication reliability decreases to ~ 97% when the normalized PLC process time increases to 0.06 - 0.28. For the trapezoidal signals an100% communication reliability is achieved for normalized PLC process time up to 0.08 (Fig. A.9c). This may be due to the plateau at the peak amplitudes of the trapezoidal signals which makes it difficult to detect delayed data points, as the transmitted and returned signals do not change in this region. The results of characterizing the communication between the Simulink model and OpenPLC using Python data transfer interface show a most probable cycle time of 100 ms. This holds true using isolated and non-isolated communication network and irrespective of type of the communicated signals, the PLC scan time, and the simulation time step. Using an isolated network slightly affected the communication reliability using the data transfer interface, with less variance in the recorded cycle time. The communication reliability increases with decreased PLC normalized process time, τ, approaching almost 100% for $\tau \leq 0.06$.

A.3. Summary

This section presents the results of research characterizing the performance of the OpenPLC runtime when integrated into the LOBO NCS platform. This research investigated the scan cycle behavior of OpenPLC and ways to achieve more consistent operation. The actual scan time of OpenPLC estimated for input scan times from 0.01 - 26 ms. Results show that the scan time within the OpenPLC is inconsistent from cycle to cycle, with periodic cycles much longer than the input scan time. The investigated subroutines determine the cause of these long scan time cycles. The modified OpenPLC source code disables the routine causing the delay, enhancing the consistency of the actual PLC scan time, and resulting in good and consistent agreements with the user specified input scan time.

The research also investigated the communication between the OpenPLC program and an external data transfer interface linking a Matlab Simulink model to the PLC. The results of the performed characterization show that the most probable communication cycle time between Simulink model and the OpenPLC using a Python data transfer interface is 100 ms. This time is the same regardless of the type of communication network used (isolated and non-isolated), the type of the generated Simulink signal, the PLC scan time, and the simulation timestep size. The communication reliability quantified as a function of the normalized PLC process time. The reliability decreases with increasing normalized PLC process time and approaches 100% when the normalized PLC process time is < 0.06. These results provide guidelines for future implementation of emulated PLCs into the LOBO NCS platform for research and investigations of cybersecurity of terrestrial nuclear power plants.