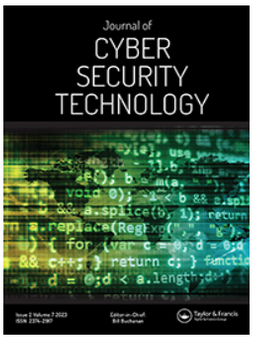

Journal of Cyber Security Technology

ISSN: (Print) (Online) Journal homepage: www.tandfonline.com/journals/tsec20

# A digital twin internal to a PLC to detect malicious commands and ladder logic that potentially cause safety violations

Aaron W. Werth & Thomas H. Morris

**To cite this article:** Aaron W. Werth & Thomas H. Morris (2023) A digital twin internal to a PLC to detect malicious commands and ladder logic that potentially cause safety violations, Journal of Cyber Security Technology, 7:2, 53-82, DOI: 10.1080/23742917.2023.2171538

**To link to this article:** https://doi.org/10.1080/23742917.2023.2171538

Published online: 01 Feb 2023.

JOURNAL OF CYBER SECURITY TECHNOLOGY
2023, VOL. 7, NO. 2, 53–82
https://doi.org/10.1080/23742917.2023.2171538

Taylor & Francis
Taylor & Francis Group



# A digital twin internal to a PLC to detect malicious commands and ladder logic that potentially cause safety violations

Aaron W. Werth and Thomas H. Morris

Department of Electrical and Computer Engineering, University of Alabama in Huntsville, Huntsville, AL, USA

**ABSTRACT**

This work presents an Intrusion Prevention System (IPS) called the Embedded Process Prediction Intrusion Prevention System (EPPIPS) to detect cyber-attacks by predicting what harm the attacks could cause to the physical process in critical infrastructure. EPIPPS is a digital twin internal to a Programmable Logic Controller (PLC). EPPIPS examines incoming command packets and programs sent to the PLC. If EPPIPS predicts these packets or programs to be harmful, EPPIPS can potentially prevent or limit the harm. EPPIPS consists of a module that examines the packets that would alter settings or actuators and incorporates a model of the physical process to aid in predicting the effect of processing the command. Specifically, EPPIPS determines whether a safety violation would occur for critical variables in the physical system. Experiments were performed on virtual testbeds involving a water tank and pipeline with a variety of command-injection attacks to determine the classification accuracy of EPPIPS. Also, uploaded programs including time and logic bombs are evaluated on whether the programs were unsafe. The results show EEPIPS is effective in predicting effects of setting changes in the PLC. EPPIPS's accuracy is 98% for the water tank and 96% for the pipeline.

**ARTICLE HISTORY**
Received 16 August 2022
Accepted 18 January 2023

**KEYWORDS**
specification-based intrusion prevention system; digital twin; SCADA

## 1. Introduction

Modern society relies heavily on critical infrastructure to function. Much of this infrastructure is computerized and managed with Supervisory Control and Data Acquisition (SCADA) systems. SCADA systems involve networks and devices, such as Programmable Logic Controllers (PLCs), servers, and other computers that make up the network's nodes. This architecture makes them similar to typical general Information Technology (IT) networks. SCADA systems mainly differ in that they interface with the physical world through actuators and sensors of PLCs and other Intelligent Electronic Devices (IEDs). Research towards

**CONTACT** Aaron W. Werth aww0001@uah.edu Department of Electrical and Computer Engineering, University of Alabama in Huntsville, Huntsville, AL, USA

mitigations of cyber-attacks has been quite extensive in the literature for SCADA systems and includes firewalls, encryption, authentication, and other techniques and technologies [1]. One type of mitigation involves Intrusion Detection Systems (IDSs) to detect malicious behavior on networks. IDSs can be used in general IT networks.

However, examples of cyber-attacks, such as Stuxnet [2] and the attack on the Maroochy [3] treatment plant, were largely able to perform their destructive acts before being discovered or detected by traditional IDSs. Stuxnet and Maroochy are both examples of zero-day attacks, because the exploits and vulnerabilities are unknown, except by the attacker [4]. The term zero-day attack refers to the fact that the cybersecurity engineer who works to patch the vulnerability has zero days to respond to the vulnerability once the attack becomes known. Because defenders do not know the zero-day attacks, mitigations to defeat these attacks from a conventional cybersecurity perspective do not exist. Therefore, it is necessary to understand how to detect and mitigate against zero-day attacks and especially those in which the effect on the physical system can be predicted before the attack has taken control of PLCs or the actuators associated with them. This work seeks to advance research toward detecting zero-day attacks against SCADA systems by presenting a generic and deployable control/command analysis Intrusion Prevention System (IPS) that takes into account how commands and ladder logic in the PLC impact the physical system. An IPS is a specific type of IDS that not only detects intrusions, but also responds to them appropriately.

It is crucial to investigate how zero-day attacks and cyber-attacks in general impact physical processes in critical infrastructure. In the literature, Hahn et al. describe a useful framework consisting of a model of a cyber-physical system and a kill chain [5]. The model consists of three main layers: (1) Cyber Layer, (2) Control Layer, (3) and the Physical Layer. When an electrical engineer or a control system engineer designs a control system for a given cyber-physical system, the engineer uses mathematical abstractions or creates a block diagram in special design software. The designs must be directly compiled or written in a programming language which will then be compiled for the actual hardware of the PLC that controls the physical process. The control layer is thus mapped to the cyber layer, and it exists in actuality as a compiled program in binary machine code form. It resides in the memory of the PLC's hardware and is processed by the CPU of the hardware. The hardware itself is connected to actuators that influence a physical process that an appropriate domain expert can describe. Sensors allow the PLC to have information about the physical process. The kill chain described in Hahn's work can be used to understand how violations of one or more of the components of cybersecurity (availability, integrity, and confidentiality), which occur in the software and networks, can lead to violations of properties of the physical process, namely stability, safety, and efficiency. Huang [6] provided seminal work investigating the physical and

economic consequences of cyber-attacks against integrity and availability in control systems. Since the threat model of this work involves malicious entities or processes attempting to change settings through commands of SCADA protocols and ladder logic uploads, it must be understood that these are attacks on integrity. In this work, the IPS serves as a form of access control to prevent integrity attacks caused by commands or ladder logic uploads that would alter the program governing the PLC's behavior. The IPS exists as an embedded module inside the PLC and serves as a digital twin of the PLC and the physical system that the PLC interacts with. The IPS is called the Embedded Process Prediction Intrusion Prevention System (EPPIPS)

The remainder of this article is organized in the following sections: Section 2 discusses relevant related works and compares them with the approach of this current work. Section 3 discusses the threat model of interest to this work. Section 4 describes the novel IPS used to detect and respond to the threat model. Section 5 discusses the overall setup used to test and evaluate the IPS with a variety of experiments. Section 6 discusses the results of those experiments. Finally, Section 7 is the conclusion.

## 2. Related works

Several related works are relevant to this research in terms of IDSs and IPSs for critical infrastructure and digital twins. McLaughlin addresses evaluating PLC code to determine if there is any potential execution path in the code that will place the cyber-physical system into an unsafe state that is defined beforehand [7–9]. What distinguishes the contributions of this research from McLaughlin's work is that EPPIPS in this research uses a model of the physical system to evaluate whether the program's actions to control the actuators would place the cyber-physical system into undesired states. In Chromik's work [10,11], a model of a power system allows an IDS in a local substation or in a central SCADA master to test a given command against a set of safety requirements. Before the SCADA system or relays in the substation process the command, the IDS determines if a violation occurs for a defined set of safety requirements for voltage and current using the model. Similarly, Lin [12,13] also created an IDS that evaluates commands to change settings in power systems. On the other hand, this work focuses on Industrial control systems other than power systems and also on commands that change settings for complex ladder logic programs. These settings may change how the program behaves. EPPIPS is capable of predicting the effects of these complex settings. In contrast, Chromik and Lin deal with only commands that more directly affect the actuators of power systems, such as the breakers. In the work by Etigown et al [3], the authors develop a similar system in principle to the previous mentioned works with a centralized system that serves as a form of access control. Certain users are permitted to control only specific variables associated with the cyber-physical

system and see only certain variables based on their assigned roles and privileges. The works by McLaughlin, Chromik, Lin, and Etigown not only detect and predict harmful commands but also can prevent their effects. Additionally, McLaughlin's work can evaluate uploaded ladder logic requests and prevent harmful ones [9]. Therefore, these works are all examples of IPSs. Also, these works have a specification indicating safety conditions. Similarly, EPPIPS is an IPS and has safety conditions in its specification. Also, the model of physical system used in EPPIPS is based on data used to train it [14].

Recent research also covers digital twins as an approach to cybersecurity for cyber-physical systems and critical infrastructure. This current work on EPPIPS is a continuation of a previous work which describes EPPIPS as an internal digital twin in a PLC [14] to make predictions of potential behavior. The previous work focuses on accuracy of a model for the physical system coupled with a modified PLC process and also the latency involved. This current work focuses on an expanded set of experiments involving normal operations, cyberattacks and malicious ladder logic uploads to evaluate how well EPPIPS can predict whether these cyberattacks impact the safety of the ICS. Other research uses digital twins for cybersecurity in various other ways. Eckhart and Ekelhart created a virtual twin of a PLC [15]. They did not create a mathematical model of the physical system in their twin. Instead, they rely on the physical system of the real world. The input or stimuli of the physical twin is taken and applied to the virtual twin based on the specification of the physical twin. Various states of the physical twin are also replicated in the virtual twin. The virtual twin is used to examine and detect malicious ladder logic and was demonstrated to be successful in the results. The virtual twin performs continuous processing in parallel with the physical twin [15]. In contrast, EPPIPS uses a model of physical system and can speed up the execution of the ladder logic to be tested. This allows for a quicker evaluation to take place, which may be necessary for making decisions with real-time applications. In [16], Gehrmann and Gunnarsson perform comparisons of system states between the physical twin and the virtual twin. Their work is similar to Echhart and Ekelhar's. Communication between the twins performed confidentially. The authors use OpenPLC for the PLC in their evaluation. EPPIPS also involved using OpenPLC. In [17], the authors deploy a digital twin in a cloud. The digital twin is intended to detect manipulation of actuator signal and sensor signals for industrial control systems. The author's approach is two-fold: first, attack detection. Second, mitigation to ensure that the ICS remains stable with reasonable performance. EPPIPS, on the other hand, uses an embedded IPS in the PLC's hardware, which may avoid much of the attack surface of a network between the PLC and the cloud. In [18], Fatemeh et al. integrate Machine Learning (ML)-based IDS with simulated attacks against digital twins. Advantages of this approach is that it used an ensemble of machine learning techniques to improve its ability to detect a variety of attacks. The digital twin, which is essentially used as a replica, is used for experiments

with cyberattacks to train the machine learning algorithms. In [19], Francia and Hall describe digital twins as a tool to perform penetration tests and assessment for an ICS. The paper makes the case that the digital twin can be constructed based on its physical counterpart and uses OpenPLC to do so. Performing the tests and assessment on the digital twin allows the researcher to avoid disrupting the physical twin it its operation.

## 3. Threat model

The threat model of this work includes malicious commands and uploaded programs that modify a PLC's behavior since these commands and programs can have a direct influence on the physical process that the PLC and the overall SCADA system manage.

### *3.1. Malicious commands*

Malicious commands are characterized by their tendency to place the physical system into an undesired state. In the context of SCADA systems, an undesired state would be an unsafe state or a state that is inefficient or has negative economic consequences [6]. This work uses predefined unsafe states, which are a part of the specification of the system as determined by a human with expert domain knowledge of various engineering systems under consideration.

### *3.2. Malicious uploaded programs*

Malicious uploaded programs in this work have an adverse impact on the physical system. Some of the malicious programs may be designed to have an immediate harmful effect. In contrast, others are designed intentionally to appear normal in their behavior initially but then have a damaging effect later. These may be termed latent effects. Such effects are desired from the perspective of the threat actor, who would like to remain hidden so that the user of the SCADA system will trust the system since it appears to be functioning normally. Two major types of malicious programs with latent effects are time bombs and logic bombs. In cybersecurity, time bombs are defined as malicious software whose malicious behavior is triggered after a given amount of time. A logic bomb is defined as malicious software that is triggered for a specific set of conditions. The concept of a ladder logic bomb can be found in Govil's work [20], which considered ladder logic or programmation for the PLC as distinct from the firmware, which is defined as the operating system (OS) and other supporting software including drivers.

## 4. Novel intrusion prevention

To counter the threat, this work presents an intrusion prevention system called EPPIPS. In its deployed operation, EPPIPS makes predictions on how a given command or program would affect the physical system. EPPIPS is a software-based module that executes inside the PLC as a stand-alone process. EPPIPS acts as a proxy between the main PLC process and the main SCADA network by intercepting all network traffic between the PLC and Human Machine Interface (HMI) and relaying all communication between the two. Figure 1 illustrates EPPIPS's design. EPPIPS consists of two major submodules: (1) the main submodule and (2) the prediction submodule. The following paragraphs explain the deployed operation of EPIPPS involving these submodules:

The main submodule intercepts network traffic sent from the HMI to the PLC. Command packets are evaluated before being sent to the PLC. The main submodule includes a shadow memory. The shadow memory stores a local copy of the contents of the PLC's memory. The contents of the shadow memory are obtained by examining the contents of *read* and *write* Modbus queries and

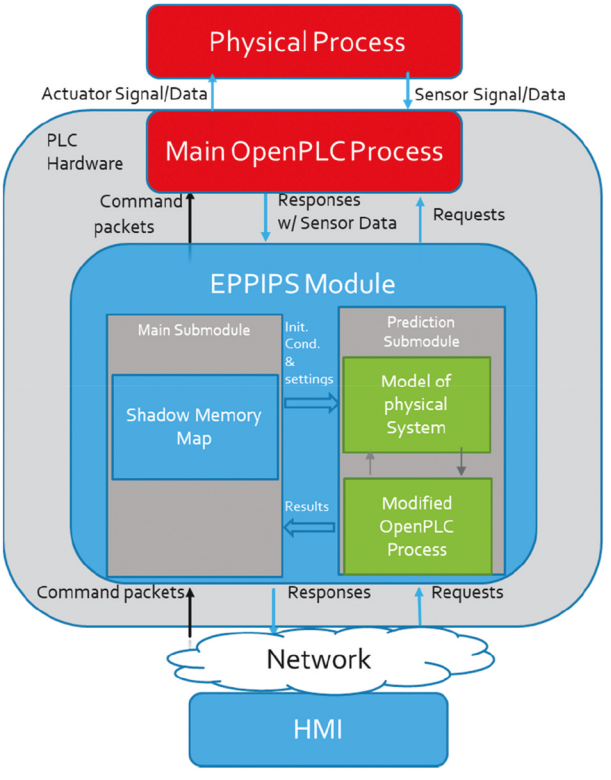


**Figure 1.** Overview of EPPIPS: an intrusion prevention system embedded within the PLC and its submodules that include (a) main submodule, which examines packet traffic and (b) prediction submodule, which is triggered when the traffic includes a command or ladder logic upload, which may in turn affect the physical process.

responses that pass through EPPIPS. When the main submodule detects a Modbus query that includes a command to change a PLC setting, the current physical system states and current settings stored in the shadow memory including those affected by the command are sent to the prediction submodule.

The prediction submodule then predicts the effect of the command using a modified version of OpenPLC (ModOpenPLC) coupled with an Autoregressive with Exogenous Input (ARX) model of the physical process. For the prediction, ModOpenPLC is run without any idle delay over a large but finite number of scan cycles as explained in the work [21]. With each iteration of a scan cycle in the prediction, EPIPPS compares the predicted process variable (PV) with a range of values as indicated by a specification. Note that the process variable is a predicted sensor value as indicated by the ARX Model, such as pressure, temperature, etc. The specification contains the safety conditions for PV as minimum (PVmin) and maximum (PVmax) values for the PV. Outside the range ($PV_{min}$ <= PV <= $PV_{max}$), the PV is unsafe. In addition, if the PV not only becomes unsafe but this unsafe event occurs at a time ($t_e$) before a minimum time ($t_{min}$), then the physical system is predicted to have imminent danger (PVmin <= PV <= Pvmax and $t_e < t_{min}$). The specification also defines this minimum time. Once the prediction submodule has performed its simulation, it forwards the result of its evaluation to the main submodule. The main submodule uses the prediction result to evaluate the command and determine if the command should be forwarded to the PLC or if the command should be dropped or delayed. In addition to one of the previous actions, EPPIPS may issue a warning. The decision on how EPPIPS responds is a heuristic process based on the predicted severity of the cyber-attacks. If a packet arrives at the PLC, and EPPIPS determines the packet to have a benign effect on the physical system, meaning the resulting behavior of the ICS would be safe and not imminently dangerous, then the packet is processed. If the resulting behavior is predicted within the prediction window to eventually become unsafe but does not cause imminent danger, then the packet is processed with an alert to the user. If however, the packet is predicted to cause imminent danger, then, the packet is dropped or delayed. The specification indicates the choice of whether to drop or delay.

For this deployed phase, as explained, a packet's arrival triggers the EPPIPS prediction submodule. However, there are other times when the EPPIPS can be triggered. Three conditions trigger execution of the prediction: (1) When EPPIPS starts intercepting communication between the PLC and HMI (2) When payload arrives, and (3) Periodically. In the first case, once the PLC starts running, and information regarding the current state of the PLC enters the EPPIPS's shadow memory, EPIPPS starts the prediction submodule. The second case is the primary case of evaluating a new command packet that EPPIPS responds to. The third case occurs

periodically to account for the fact the prediction window of EPPIPS is limited.

The current methodology and design of EPPIPS is a more developed version of the approach introduced in a previous work [22] and is described in detail in another more updated previous work [21]. This current work focuses on EPPIPS as it runs in its deployed operation and how well it can respond to cyberattacks and complement safety equipment present in critical infrastructure.

As EPPIPS runs in its deployed operation, another device, a Safety Instrumented System (SIS), is typically also used for safety purposes in many ICSs. SISs are systems whose purpose is to monitor the behavior of the physical process/system under the control of the PLC. An SIS acts to place the physical process into a safe state when the physical process enters into or closely approaches an unsafe state. In this context, the state is a process variable of the physical system such as temperature or pressure which may be at a dangerous level. The SIS is independent of the PLC and may have actuators and sensors separate from the PLC's actuators and sensors [23]. EPPIPS is not the same as an SIS, although both deal with unsafe conditions. EPPIPS can predict what an incoming packet with a given payload will do to the physical system. The arrival of the payload is a cyber event that EPPIPS can identify, respond to, and log. Being able to attribute a predicted unsafe condition to a given packet is helpful for performing forensics on the SCADA system. On the other hand, the SIS responds to a situation where the process variable is about to become unsafe by taking actions through its own actuators to alleviate the situation. The SIS responds regardless of whether the unsafe situation is caused by a cyber event or a physical fault. A fault may happen when a valve or pump becomes unresponsive in a given physical system.

Although the SIS may ensure that the physical process can be brought to a safe state when approaching an unsafe state, cyber-attacks may still have adverse effects on the ICS despite the use of an SIS. The act of an SIS system responding to impending safety concerns places the ICS essentially out of commission from operating as designed for normal conditions. An attacker may wish to take advantage of the disabled operation to sabotage the industrial control system. In this case, sabotage means that the SCADA system or ICS cannot perform useful work since the SIS effectively shuts down the ICS. The attacker can perform the sabotage by sending commands or uploading new ladder logic to drive the physical system toward the unsafe state to manipulate the SIS to act. If an SIS exists but EPPIPS is not present in an ICS while a cyber-attack induces an unsafe condition, the SIS will act to place the system into a safe state as the SIS was designed to. However, once this has occurred, plant personnel may try to reset and restart the ICS when in fact, the ICS is still compromised. Furthermore, since the SIS is likely to be a PLC just like the controller for the physical process, the SIS is not immune from being hacked itself in the same ways as the controller. In fact, Triton also known as Trisis, is a recent example of malware against an SIS in a petrochemical facility [24].

**Table 1.** Comparison of major characteristics of EPPIPS and SIS in terms of safety and cybersecurity.

| EPPIPS | SIS |
|---|---|
| EPPIPS identifies and responds to *cyber events* (i.e. arrival of packet) predicted to cause safety concerns | SIS identifies and responds to *physical events* affecting safety regardless of their cause. |
| EPPIPS can attribute a cyber event's potential physical impact to the cyber event. | SIS cannot attribute a cyber event's potential physical impact to the cyber event. |
| EPPIPS can respond to a cyber event *before* it has an impact by blocking, delaying, or warning. | SIS can only respond to a cyber event *after* the event has a physical impact by shutting down the ICS. This means a cyber-attack can sabotage the ICS. |
| EPPIPS can protect the ICS from safety concerns related to cyber threats, but EPPIPS is best augmented with an SIS. | SIS is useful to prevent physical destruction, even if a cyber-attack can take advantage of an SIS in other ways. |

However, the SIS is usually physically isolated from the main ICS network, reducing the SIS's attack surface.

In summary, both the EPPIPS and the SIS are useful devices that manage safety concerns. These devices have similarities and differences. EPPIPS is equipped to identify and respond to cyber-events that trigger physical events. On the other hand, an SIS can ultimately handle safety concerns, but cyber-attacks can potentially manipulate the SIS. Also, the SIS as an embedded system is not immune to cyber-attacks in general. Together, both devices may be used to improve the overall safety and security of the ICS. Table 1 below highlights the significant characteristics of the two devices in terms of safety and cybersecurity.

## 5. Experimental setup

Performing experiments in this work involves virtual testbeds to represent ICS. Therefore, this research uses the framework for an ICS fully explained by Alves et al [25]. in 2018. The framework divides an ICS logically into five components. The five components as seen in Figure 2 are (1) the physical system, (2) the cyber-physical link, (3) the PLC, (4) the network, and (5) the Human Machine Interface (HMI) for remote monitoring and control. A hypervisor runs virtual machines representing the HMI and PLC, which make up the nodes of the ICS network. The rogue device and SIS were also implemented and used.

The experimental scenarios that run on the setup are based on situations that an ICS can experience with variations and combinations of these situations. The experiments were chosen to test the versatility of EPPIPS in detecting a variety of attacks affecting different settings at various times in the simulation and in cases in which the physical system is in different states. The primary purpose of running these experimental scenarios is to test the accuracy of EPPIPS. The scenarios divide into two broad categories of experiments that will be conducted: (1) Normal Conditions and Operations and (2) Cyberattacks. The first category consists purely of normal operations. The second major category, which involves attacks overlaying normal operations, consists of five

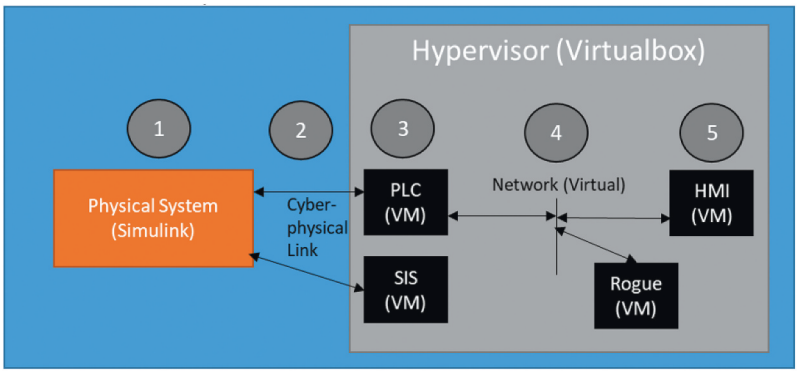


**Figure 2.** Illustration of setup for experiments: testbed includes a physical system, cyber-physical link or wired connection, PLC and SIS as embedded devices, network for TCP/IP communication, a rogue device and an HMI.

subcategories: (a) High Setpoint Attack, (b) Switch to PI (proportional/integral) Control with Random Settings, (c) Switch to Manual Mode with Pump on, (d) Targeted Attack from Rogue, and (e) Targeted Attack from HMI. The attacks choose some of the values for their settings at random. The randomization uses a uniform distribution so that the space of possible parameters for the attacks has relatively even coverage. The following table summarizes these scenarios types used in the experiments (Table 2):

The high setpoint attack involves the rogue VM first sending request packets to obtain the value of the process variable of the system and then injecting commands to the PLC to change the high setpoint. The rogue VM randomly selects the value for setting the high setpoint register and launches the attack at a random point in time during the simulation. Although this attack seems simple, it is chosen to test EPPIPS's ability to transfer important status information of the ladder logic variables, such as the state of the pump and also the accuracy of prediction when compared with the actual behavior.

**Table 2.** Scenarios types involving normal and various complex injection attacks.

| Category | Subcategory | Registers Affected |
|---|---|---|
| Normal | Automatic | Unaffected |
| Attack | High Setpoint Attack | High Setpoint (%QW8) |
| | Switch to PI Control with Random Settings | Mode register (%QW4)<br>P Gain Register (%QW1)<br>I Gain Register (%QW2)<br>Reference Register |
| | Switch to Manual Mode with Pump on | Mode register (%QW4)<br>Pump Register (%QW5)<br>Intensity (%QW6) |
| | Targeted Attack from Rogue | Mode register (%QW4)<br>Pump Register (%QW5)<br>Intensity (%QW6) |
| | Targeted Attack from Compromised HMI | Mode register (%QW4)<br>Pump Register (%QW5)<br>Intensity (%QW6) |

The attack to switch to PI control with Random Settings is also an attack the rogue device carries out. This attack involves injecting a command to change multiple registers simultaneously. The affected registers are the mode register, the P gain register, the I gain register, and the reference register. The attack changes the mode register to 3 to enter into PI control mode. The attack also randomly selects values for the P and I gains. The rogue device launches the attack at a random time during the simulation.

The attack to switch to manual mode and turn on the pump in this work uses an injected command to change multiple registers, specifically three for this attack. One register affected is the mode, which is changed to 1 to cause the ladder logic program to enter into manual mode. Another register is the pump register, which turns on the pump when the register is equal to 1. The third is the register for the intensity, which is a randomly chosen value.

The targeted attack from the rogue device is an attack intended to inflict damage on the ICS. This attack uses an auxiliary reconnaissance attack to determine the state of the process variable periodically. When the process variable reaches a given value as the ladder logic program is functioning as normal, then the targeted attack will act similarly to the attack to change to manual mode and turn on the pump. However, in this case, the targeted attack will choose the maximum value for the pump's intensity setting.

The targeted attack from the HMI is for representing the situation in which malware or a malicious user compromises the HMI. At a random time in the simulation, this attack launches an attack to change multiple registers so that the PLC enters manual mode and turns on the pump at the highest intensity. The purpose of this attack is to demonstrate that the HMI, a trusted system, can be the node through which the attack occurs against the PLC.

Because of the number and complexity of scenarios and the need for precise timing for when attacks occur, it was necessary to create automation scripts to run from the host machine that the hypervisor runs on. A master automation script resets the PLC's main process and directs the rogue virtual machine used in the experimental setup to behave appropriately for a given scenario. For example, the script signals the rogue device when to launch the scenario and which of the 6 types of scenarios to use. The scenario types are enumerated from 0–5, and the script randomly chooses one of the values to determine what scenario type to launch. The script also specifies what settings to use for the attack. The master automation script runs many iterations of each of the main categories of scenarios described above. The script systematically runs through a large number of these scenarios in order to achieve results that determine the effectiveness of EPPIPS. For each scenario that runs, the 'ground truth' is established from the MATLAB simulation as to whether the scenario becomes safe or unsafe due to a command packet. Also, the sensor and actuator data are recorded in the ground truth over time. In addition, within each scenario, EPPIPS makes a prediction based on the command packet that arrives at the PLC. This

prediction exists as time-series data and may be compared with the ground-truth data to evaluate EPPIPS. The time series data consist of the predicted sampled sensor data over time and the predicted sampled actuator state over time. These data are logged by EPPIPS and can be exported as CSV data. Furthermore, the data from the ground truth is recorded in MATLAB and may also be exported and compared with the data logged by EPPIPS for later analysis.

## 6. Results

To evaluate the effectiveness of EPPIPS, experiments were performed on virtual testbeds that include a water tank and a pipeline, both of which use variable-speed pumps. These experiments consist of command-injection attacks and ladder logic uploads.

### *6.1. Evaluation of classifying incoming payloads*

To evaluate how well the IDS classifies the incoming command packets to diagnose whether they are harmful, it is necessary to compare the predicted outcome with the outcome that occurs for the 'ground truth'. The ground truth is the true condition and behavior of the physical process when an experiment is run. Note that when conducting the experiments, all incident response actions by EPPIPS besides processing packets are suspended, which means that payloads predicted to be harmful are not dropped or delayed. This allows the payload to be processed and to have an effect on the testbed so that the comparison can be made with the prediction.

In this work, classifications are made according to two main schemes: (1) safe vs. unsafe and (2) no imminent danger vs. imminent danger. The first scheme involves classifying the incoming packet as safe or unsafe. The term *safe* in this work means that the process variable remains within the safety conditions defined in the specification. The term *unsafe* means the process variable violated the conditions. Classifications are also made in this manner for whether a given scenario is imminently dangerous or not imminently dangerous. *Imminent Danger* means the process variable becomes unsafe within a predefined amount of time given in the specification, whereas *No Imminent danger* means that the process variable remains safe in the given time period. However, in the case of no imminent danger, the process variable may be safe or unsafe after the time period.

Evaluating the IDS's ability to make classifications for an individual experiment involves comparing the prediction of EPPIPS in that experiment to the ground truth data of the experiment. If the IDS predicts that a command will cause the physical system to become unsafe when, in reality, according to the ground truth data, the physical system remains safe, then this is called a *false*

*positive* (FP) in detecting the unsafe condition. On the other hand, if the IDS predicts a safe condition and the physical system actually becomes unsafe, then this constitutes a *false negative* (FN). When the command is correctly classified as unsafe, this constitutes a *true positive* (TP). A correctly classified safe scenario is a *true negative* (TN). These same concepts in comparing the ground truth with the prediction also apply to imminent danger. When many scenarios are run with command packets arriving at the PLC, EPPIPS will classify them. A measure of accuracy that represents all the experiments can be made concerning EPPIPS. This measure is termed *classification accuracy*. Classification accuracy is defined as the number of correctly classified scenarios divided by the total number of scenarios. Classification accuracy is expressed using the mathematical notation in Equation 1:

$$Classification\ Accuracy = \frac{Correctly\ Classified\ Scenarios}{Total\ Scenarios} = \frac{TP + FN}{TP + TN + FP + FN} \tag{1}$$

This approach of evaluating the accuracy of the classification concerning safety is based on Lin's work [12,13].

For the experiments in classification, results were achieved by running the master automation script described in Section 4. The script runs a large number of scenarios for the two related testbeds – the pipeline with Variable Speed Pump (VSP) and the storage tank with VSP. During these scenarios, EPPIPS makes predictions starting from when the attack occurs to the end of the scenario. The scenarios are each 00 seconds for the pipeline, and the scenarios are 00 seconds for the storage tank since the storage tank has a much slower response. The results are described overall and by scenario type. Confusion matrices are given for each of the testbeds in addition to the classification accuracies. Selected Scenarios including attacks and normal operations are described with plots in the following subsections for the two related testbeds.

The breakdown in terms of the number of scenarios by type for both testbeds is seen in Table 3. Pie charts in Figure 3 illustrate the percentage breakdown of the scenarios by type for the testbeds.

Table 4 and Table 5 show the confusion matrices for the pipeline with VSP as a result of running 100 experiments. The classification accuracy for safety was 98%, and the classification accuracy concerning imminent danger was 98%.

**Table 3.** Number of scenarios by type in related testbeds – pipeline and storage tank.

| Code | Scenario Type | Pipeline with VSP | Storage Tank with VSP |
|---|---|---|---|
| ST0 | Normal Operations | 19 | 11 |
| ST1 | High Setpoint Attack | 21 | 13 |
| ST2 | PI Control Attack | 18 | 17 |
| ST3 | Manual Mode Attack | 17 | 23 |
| ST4 | Targeted Attack from Rogue | 11 | 21 |
| ST5 | Targeted Attack from HMI | 14 | 15 |

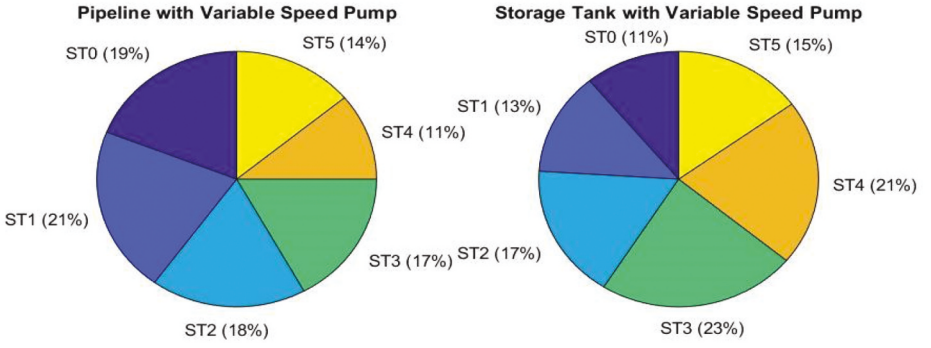


**Figure 3.** Pie chart representation of scenarios for both related testbeds by type, which is indicated by the code term from Table 3.

The storage tank with variable-speed pump was also used for experimentation with 100 scenarios. The resulting confusion matrices are produced below. Table 6 is the confusion matrix that shows the performance of classifying the scenarios with the storage tank as safe or unsafe. The classification accuracy for safety was 96%. Likewise, Table 7 shows the performance in classifying imminent danger. The classification accuracy for imminent danger was 100%.

Table 8 shows a summary of the major statistics for classification. These statistics were calculated from the data in the confusion matrices above. The accuracy is reasonably close to 100% for both the Pipeline and Storage Tank. The precision is 100% meaning that, of the experiments performed, all of EPPIPS

**Table 4.** Confusion matrix for classifying pipeline scenarios - safe vs. unsafe.

| | | Predicted | |
|---|---|---|---|
| | | Unsafe | Safe |
| Actual | Unsafe | 43 (TP) | 2 (FN) |
| | Safe | 0 (FP) | 55 (TN) |

**Table 5.** Confusion matrix for classifying pipeline scenarios - imminent danger.

| | | Predicted | |
|---|---|---|---|
| | | Imminent Danger | No Imminent Danger |
| Actual | Imminent Danger | 39 | 2 |
| | No Imminent Danger | 0 | 59 |

**Table 6.** Confusion matrix for classifying storage tank scenarios - safe vs. unsafe.

| | | Predicted | |
|---|---|---|---|
| | | Unsafe | Safe |
| Actual | Unsafe | 50 | 4 |
| | Safe | 0 | 46 |

**Table 7.** Confusion matrix for classifying storage tank scenarios - imminent danger.

| | | Predicted | |
|---|---|---|---|
| | | Imminent Danger | No Imminent Danger |
| Actual | Imminent Danger | 0 | 0 |
| | No Imminent Danger | 0 | 100 |

predicted unsafe situations are actually unsafe. However, EPIPPS sometimes classifies unsafe situations as safe. Therefore, the recall is not always 100%. The F1-Measure, another type of measure of accuracy, for the two ICSs is also reasonably high but not always perfect. The Matthews Correlation Coefficient (MCC) is a type of correlation for binary values. In this case, the binary values would be safe or unsafe. The MCC indicates how correlated the predicted versus the actual are for the scenarios. High values are desired for MMC and the table indicates reasonably high values. The reason for calculating all of these statistics can be to gauge how well the IPS performs, but mainly it is to serve as a basis of comparison with other similar IPSs.

For these sets of scenarios above, the misclassified cases are noteworthy. The misclassified cases for the pipeline were false negatives. EPPIPS also had two false negatives when classifying imminent danger versus danger for the pipeline. The misclassified cases for the water tank are false negatives. The implication of having a false negative is that EPPIPS will potentially treat the incoming command as if it is harmless when it is harmful. There are four false negatives when classifying safe versus unsafe for the storage tank. There were no misclassified cases for the storage tank when classifying imminent harm because the water tank has a much slower response when compared to the water tank. For the misclassified cases of the Storage tank, the predicted behavior falls short of the max safety threshold However, the actual behavior does cross the threshold. This is due to some nonlinear behavior in the storage tank and to the linear behavior of the prediction. This becomes apparent when the pump is not running at its full intensity.

Also, representative scenarios of the two related testbeds – the storage tank and the pipeline – are also studied. These representative scenarios are for the major scenario types explained in Section 4. Figure 4(a–f) depicts selected example scenarios representing the scenario types for the pipeline with variable speed pump. Likewise, Figure 5(a–f) shows example scenarios for the storage tank with variable-speed pump. These scenarios for both testbeds are (a) normal, (b) high-setpoint attack, (c) PI control attack, (d) manual mode attack,

**Table 8.** Summary statistics for pipeline and storage tanks scenarios.

| Industrial Control System | Classification Scheme | Accuracy | Precision | Recall | F1-Measure | MCC |
|---|---|---|---|---|---|---|
| Pipeline | Safe vs. Unsafe | 0.9800 | 1.0000 | 0.9556 | 0.9773 | 0.9602 |
| | No Imminent Danger | 0.9800 | 1.0000 | 0.9512 | 0.9750 | 0.9592 |
| Storage Tank | Safe vs. Unsafe | 0.9600 | 1.0000 | 0.9259 | 0.9615 | 0.9230 |
| | No Imminent Danger | 1.0000 | - | - | - | - |

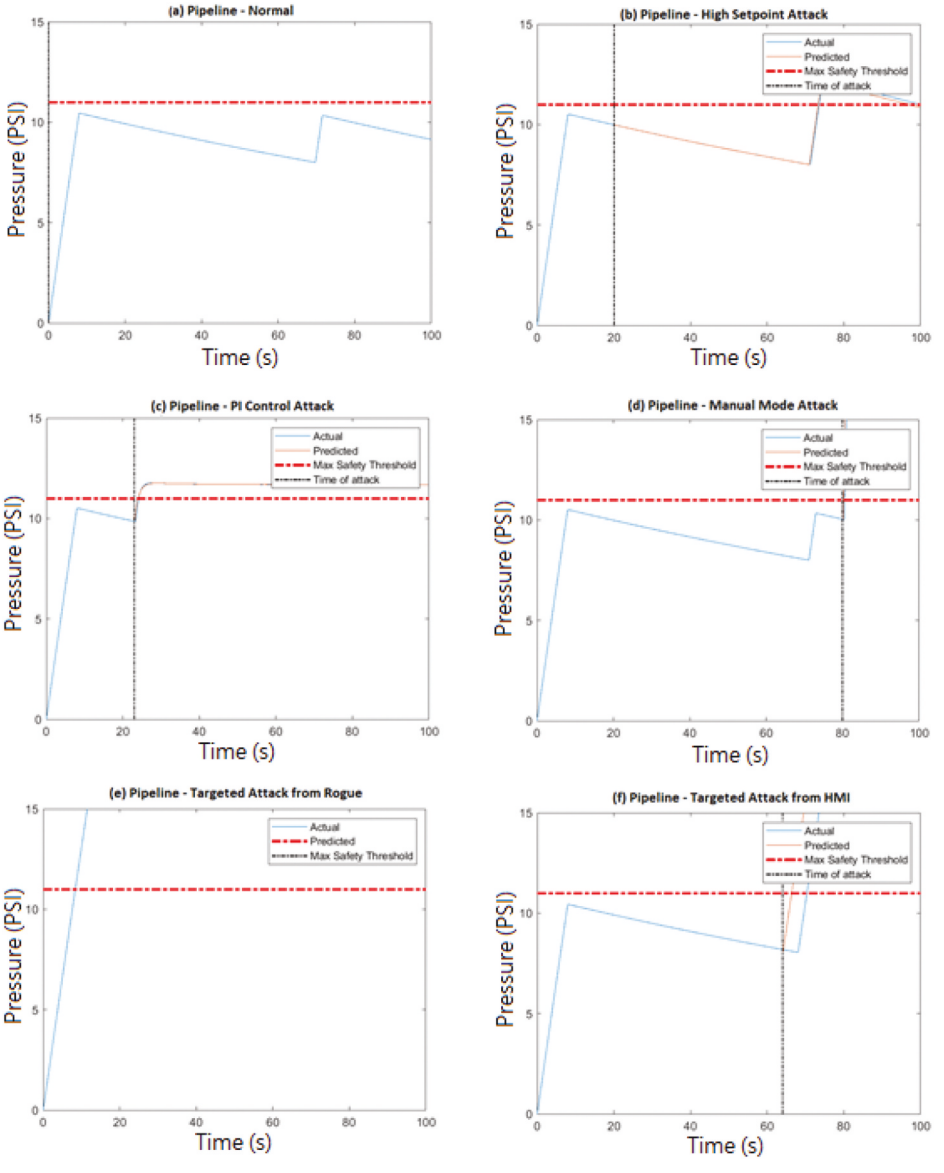


**Figure 4.** Pipeline with VSP - selected examples from each scenario type.

(e) targeted attack from rogue, (f) Targeted attack from HMI. The purpose of the selected scenarios is to provide examples and explanations for scenarios types in how they typically behave.

### *6.1.1. Normal operations*

Figure 4(a) depicts the pipeline with VSP under normal operations. Likewise, Figure 5(a) shows the storage tank with VSP under normal operations. Ladder logic programs in hysteresis mode are controlling both physical processes so that they possess safe behavior with a given load. The blue curves in the figures

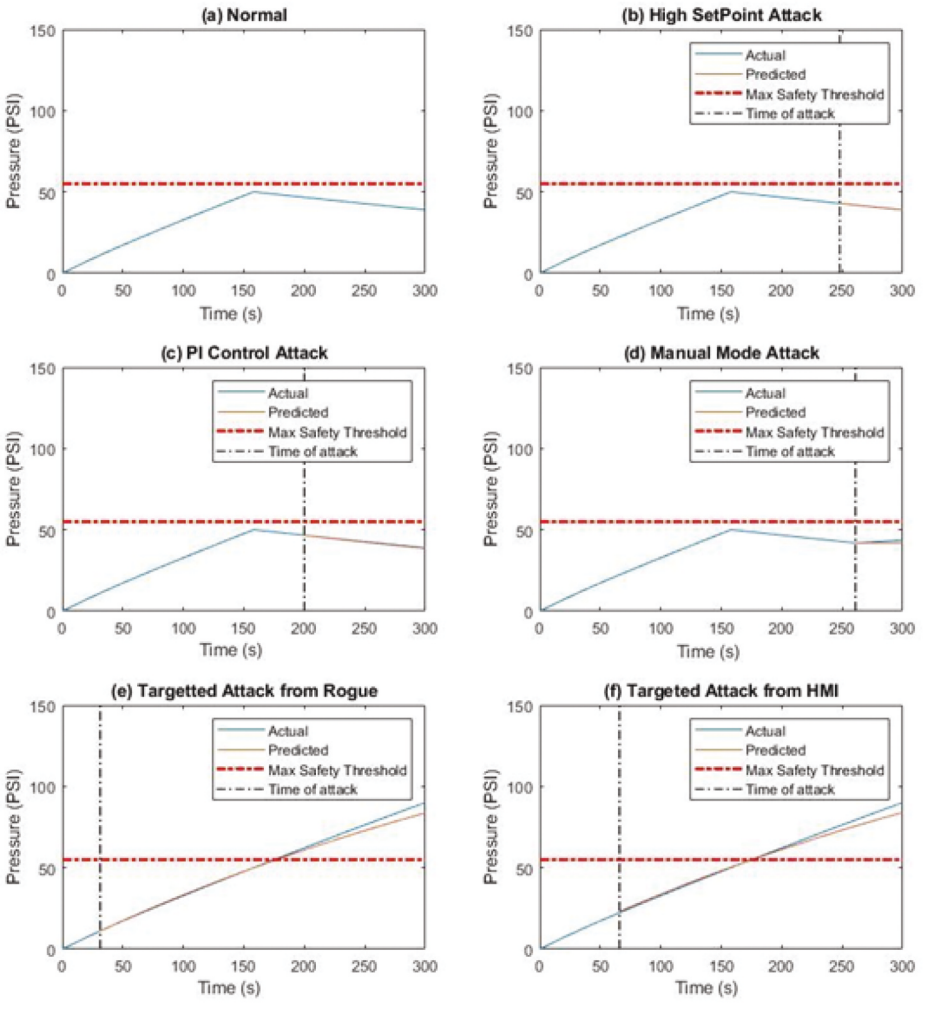


**Figure 5.** Storage tank with VSP - selected examples from each scenario type.

represent the true behavior of these testbeds over time. Note that all of the other scenario types assume that the ICS is under these normal operations until the attack occurs.

### *6.1.2. High setpoint attack*

Figure 4(b) shows the pipeline undergoing an attack in which the rogue device sends a Modbus command packet to change the high setpoint randomly. In this case, the high setpoint is selected to be above the maximum safe state. For the pipeline, the attack occurs 0 seconds into the experiment and the new setting for the high setpoint is 11.94 PSI. Both the blue and the red curves that represent the actual and predicted behavior eventually cross the maximum safety threshold. This attack is also performed for the storage tank with VSP in Figure 5(b). For the storage tank, the attack occurs 48 seconds into the

experiment and sets the high setpoint register to 52.5%, which does not cross the threshold.

#### 6.1.3. PI control attack

For this attack from the rogue device, as seen in Figures 4(c) and 5(c), the PI mode is selected and the reference or setpoint is chosen to be above the maximum safe state. Also, the gain parameters are changed for the control system. The rogue device makes all of these changes using a single command for changing multiple registers in Modbus. The attack occurs against the pipeline seconds into the experiment. The P gain is set to 68, and the I gain is set to .005. The reference is set to 11.68 PSI. As can be observed in Figure 5(c), the process variable goes to its steady state, which is the reference setpoint. For the storage tank, the attack occurs 3 seconds into the experiment. The P gain is set to 51. The I gain is set to .007, which is 7 times higher than it usually is. Also, the reference is set to 13.31% which is far lower than the safety threshold. In Figure 5(c), actual and predicted behavior slope downward as expected since the PI controller is designed to reach setpoint, which is much lower than the process variable (the pressure) when the attack first begins.

#### 6.1.4. Manual mode attack

This attack involved the rogue device changing the mode to manual mode, turning on the pump, and randomly selecting an intensity level for the variable-speed pump. For the pipeline example in Figure 4(d), the attack occurred 0 seconds into the experiment at a high intensity for the pump. For the storage tank example in Figure 5(d), the time the attack occurred was 61 seconds into the experiment. The intensity level was randomly set to 22.4%.

#### 6.1.5. Targeted attack from rogue

In this attack, the rogue device uses the auxiliary attack to query the PLC for information concerning the state of the process variable. Once the process variable reaches a value determined by the rogue device, the rogue turns on the pump and chooses the maximum setting for the pump. Figure 4(e) depicts this attack for the pipeline with VSP. When the process variable equals 9.65 PSI or more in the PLC's normal operation, the attack turns on the pump at full speed. Figure 5(e) depicts the attack for the storage with VSP. For the storage tank, the attack starts when the process variable is equal to or greater than 11.1%.

#### 6.1.6. Targeted attack from HMI

In this case, the HMI is assumed to have been compromised and the malicious software on the HMI launches the attack at a random time. When the HMI launches the attack, the HMI sends a command to change the PLC to manual mode, turn on the pump with the maximum setting for the speed. Figure 4(f)

depicts this attack for the pipeline with VSP, which randomly occurs 4 seconds into the experiment. Figure 5(f) shows the attack for the storage tank with VSP that launches 6 seconds into the experiment. As a result of the attacks, the actual behavior moves upward rapidly for both testbeds.

The results of these scenarios were presented numerically and graphically to allow for understanding and comparison between the two related testbeds and to evaluate the performance of EPPIPS in classifications. From the results seen in the confusion matrices, the values for accuracy were reasonably high. The accuracy in classifying safe versus unsafe for the pipeline is 98%. There were only a few cases that are misclassified. The accuracy in classifying with regard to imminent dangers was 98% for the pipeline. For the storage tank, the accuracy was 96% in classifying with respect to safety, and the accuracy was 100% in classifying with respect to imminent danger. The classifying capability of EPPIPS was primarily able to perform as intended because of the high accuracy rate. Researchers typically desire to approach an ideal accuracy of 100% for intrusion detection systems, even if the ideal accuracy is not always achievable in the real world. Because of this, it is vital to understand the issues that cause the misclassification in order to mitigate against the misclassification. The misclassified cases are generally due to slight modeling error or inaccuracy of the models when the experiment causes the process variable to be close to the unsafe region. Because of this potential issue, it may be advisable to have a region below the unsafe region called a 'high-risk' region. If the prediction reaches the high-risk region, then EPPIPS should flag this and treat the situation as if it is unsafe.

### *6.2. Evaluation of ladder logic uploads*

EPPIPS is also capable of assessing ladder logic uploads. Before the main PLC process receives the new ladder logic, EPPIPS predicts how the ladder logic would cause the cyber-physical system to behave given the ladder logic's default settings for a number of scan cycles. This can be considered a pre-screening to ensure that there are no time bombs within that window of clock cycles. However, if the malicious portion of the ladder logic code only becomes active with specific settings or states of registers, EPPIPS can diagnose the situation only if an event is about to trigger the malicious part of the ladder logic. The following plots depict the predictions of EPPIPS when evaluating a normal ladder logic program (Figure 6(a)), a malicious ladder logic program with a time bomb (Figure 6(b)), and a malicious ladder logic program with a logic bomb (Figure 6(c)). The normal is seen in the pre-screening evaluation as benign. The time bomb is detected in pre-screening evaluation since the malicious element of the time bomb occurred within the prediction window. The ladder logic bomb is triggered when the user or a computer process sends a command to the PLC to change the mode of operation to PI Control. The

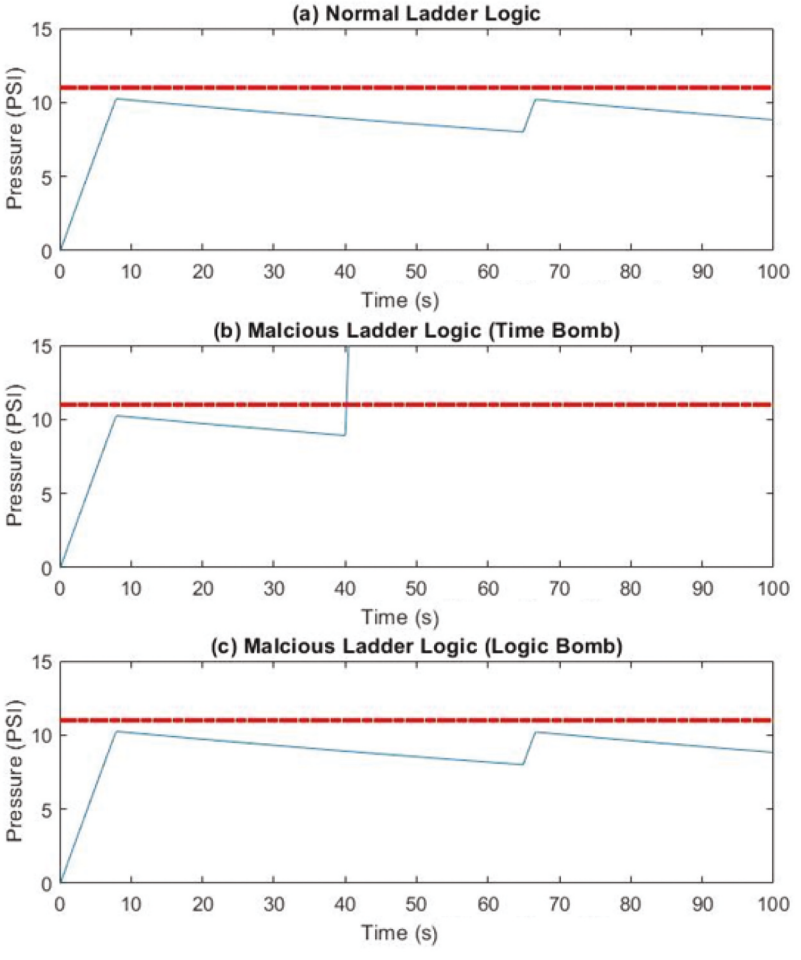


**Figure 6.** Predictions from prescreening ladder logic uploads in the pipeline testbed: (a) normal ladder logic, (b) malicious ladder logic (time bomb), (c) malicious ladder logic (logic bomb). Note that blue in the plots are the actual behavior of the pressure over time. The red dashed line is maximum safety condition.

default setting before any change of the mode by a user or command is hysteresis mode in the ladder logic. Hysteresis mode maintains the pipeline pressure between two setpoints.

For the case of the program with a logic bomb, it is not clear that the ladder logic is benign. Evaluation of the ladder logic by prescreening the ladder logic did not detect the presence of the logic bomb. Only when the ladder logic has been loaded to the PLC can the determination be made. The figure below shows behavior of the physical system controller by the PLC when running the logic bomb. Figure 7(a) shows the logic bomb without any external command triggering the bomb. Figure 7(b) shows the logic bomb with a command sent to trigger the bomb. In this scenario, a prediction is made by EPPIPS when the command comes to the PLC.

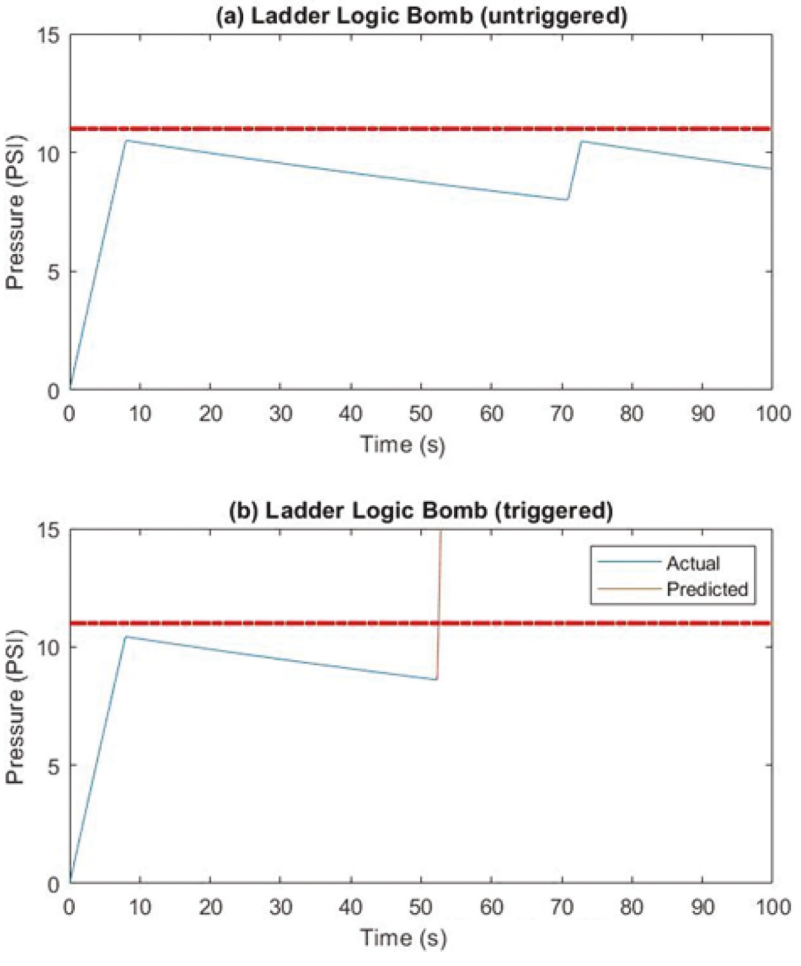


**Figure 7.** Malicious ladder logic (logic bomb) loaded to PLC: (a) ladder logic bomb is not triggered by any external event. (b) ladder logic triggered by external event roughly 55 seconds into simulation.). Note that blue in the plots are the actual behavior of the pressure over time. The red dashed line is maximum safety condition.

### *6.3. Incident response and other related case studies*

Besides being able to diagnose payloads concerning safety and immediate danger, EPPIPS can respond to them also. Several experiments were performed to observe how EPPIPS responds in various conditions. These experiments include normal conditions in Figure 8(a), manual mode attack with a low-intensity setting in Figure 8(b), and manual mode attack with a high-intensity setting in Figure 8(c). The first case is benign. In the second case, which had the low-intensity setting, EPPIPS allows the PLC to process the command packet even if the process variable eventually becomes unsafe, it is not imminently dangerous. On the other hand, if the EPPIPS predicts imminently dangerous

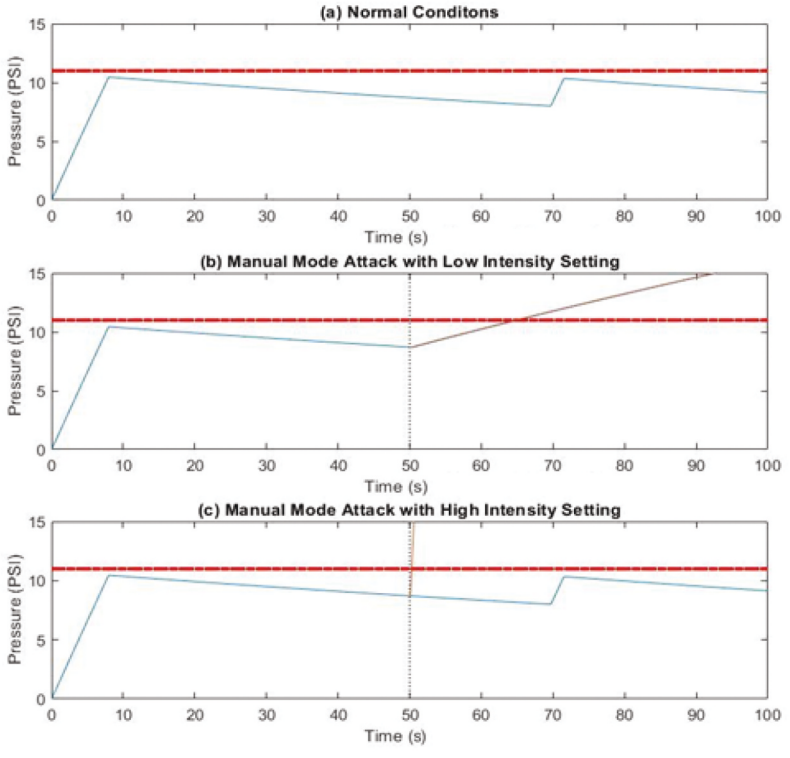


**Figure 8.** Incident response tests under varying conditions: (a) normal conditions with no command sent, (b) a change setting caused by a command, (c) another change of setting involving high intensity. Note blue represents the actual behavior. Orange is predicted behavior. The red dashed line is the maximum safety condition.

behavior as in Figure C, then it will drop the command packet. For the latter two cases, the commands are sent 0 second into the experiments.

In addition to these incident response cases, a scenario is run where the pipeline begins in normal operation and an attack occurs. This is done to observe how EPPIPS responds but also how the attack appears on the network. Figure 9 shows a Wireshark log of packets between two nodes in the network: the PLC and the HMI.

One of the packets originated from the HMI and changed the settings for the PLC using the 'Write Multiple Registers' Modbus command (highlighted in blue in Figure 9). The overall traffic and this command appear as valid commands. However, the command caused adverse behavior to the ICS (Figure 10). EPPIPS can detect this seemingly normal command and predict its effect accurately as shown in the figure where the predicted behavior very closely matched the actual behavior

Also, a final case study was examined in which the Safety Instrumented System (SIS) is added to the testbed, and the pipeline with variable-speed

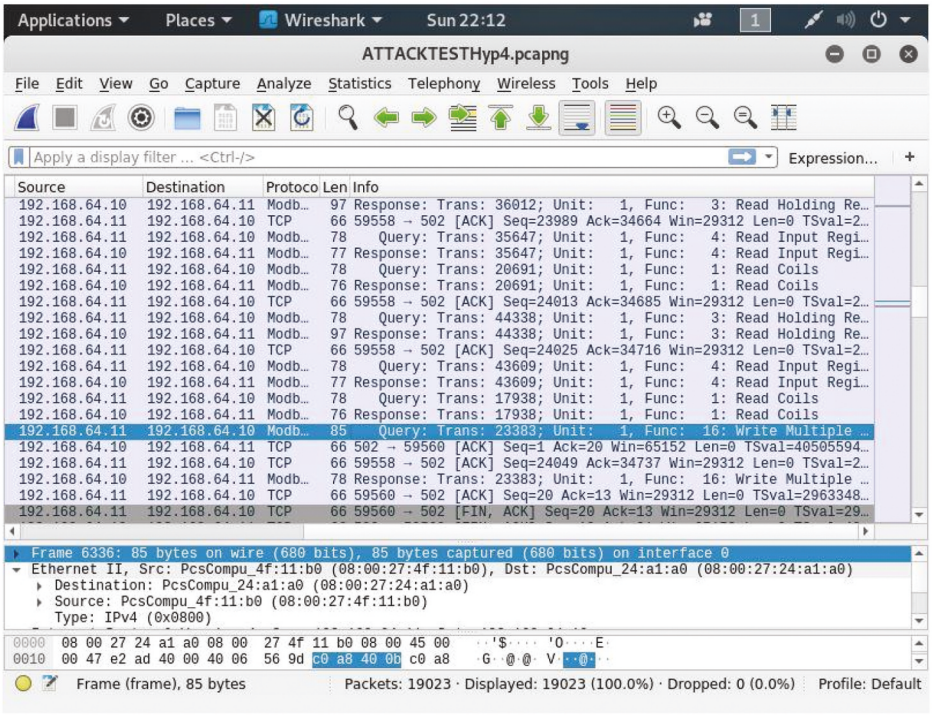


**Figure 9.** Wireshark log of packets during the operation of the pipeline testbed: command packet of interest highlighted in dark blue.

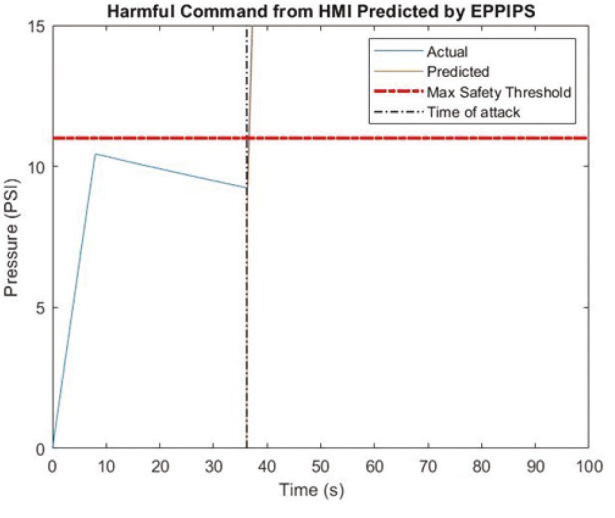


**Figure 10.** Effect of harmful command from HMI, where the predicted behavior accurately represents the ground truth behavior.

pump is running with normal operation in hysteresis mode. An attack occurs slightly after 6 seconds into the simulation to set the PLC to manual mode and turn on the pump to 10% of maximum setting for the speed. When the pressure reaches a defined threshold in the SIS, which is equivalent to the maximum safety threshold, the SIS will disengage the pump from running completely. The following plot illustrates this scenario (Figure 11).

In Figure 11 above, EPPIPS predicts the behavior on the process variable that the attack induces. The reason that the actual behavior of the process variable goes downward after the PV crosses the max safety threshold is that the SIS has taken action to counteract the effect of the attack. The reason that this occurs somewhat beyond the max safety threshold is that the pipeline has some delay in its response.

Also, the command injection attack was performed similarly against the pipeline, but this time the attack turned the pump on to its full speed (100%). Figure 11 illustrates the scenario below. When the SIS counteracts the PV going beyond the threshold, the PV rises farther than in the case of the attack running the pump at 10%.

Using the SIS is helpful to avoid unsafe states and ultimately harm to the ICS. However, the ICS is shut down and is kept from doing useful operations. As seen in Figures 11 and 12, EPPIPS can predict how the command will behave. EPPIPS is also able to attribute the command packet as causing the issue. When plant personnel investigate the incident, they can use EPPIPS to identify whether the source of the problem was attributed to a cyber event. Knowing the nature of the issue aids them in resetting the ICS so that the issue can potentially be avoided in the future.

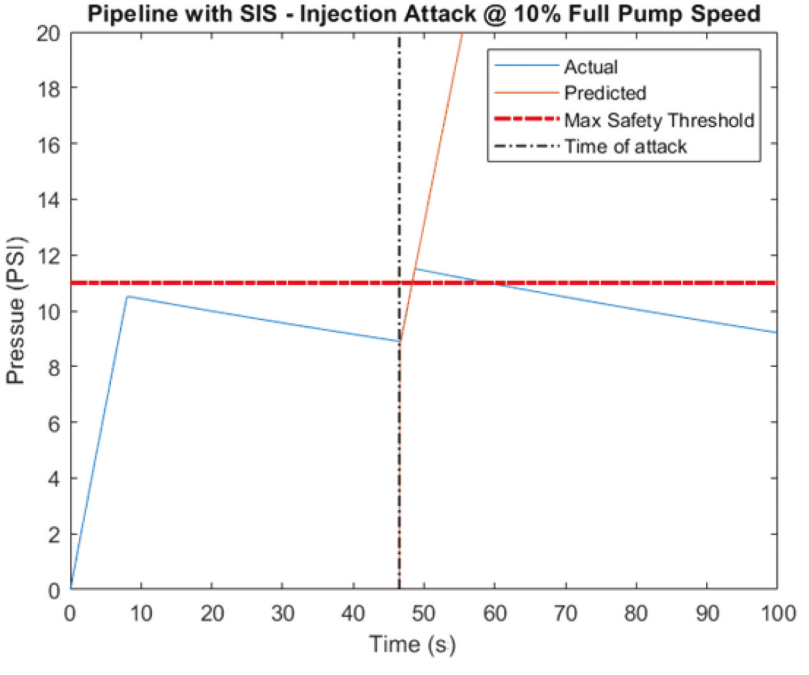


**Figure 11.** Pipeline scenario with SIS undergoing attack (pump at 10% speed).

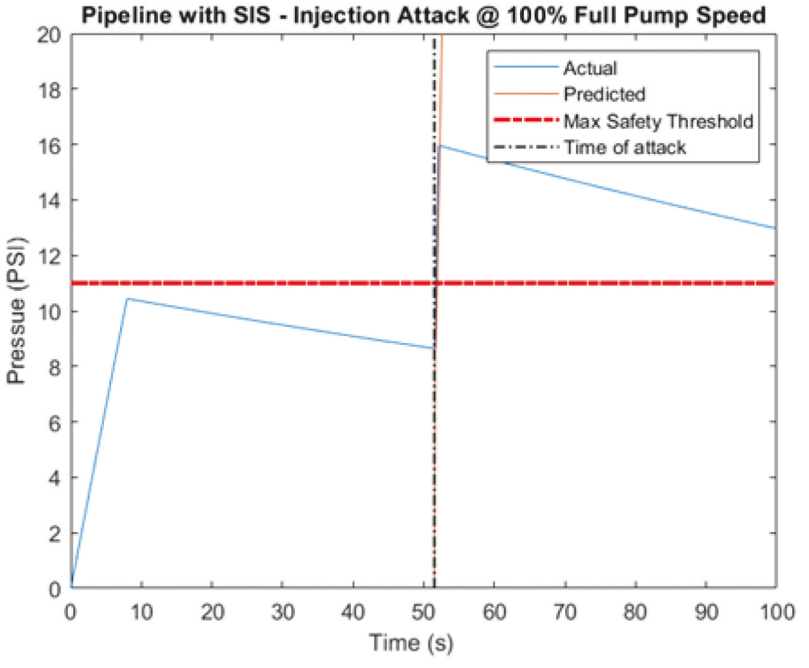


**Figure 12.** Pipeline scenario with SIS undergoing attack (pump at 100% speed).

### *6.4. Overall discussion and analysis*

The large sets of experiments involving the pipeline with VSP and storage tank with VSP demonstrated variations of attacks that could drive the ICS to unsafe states. A typical PLC connected to a network has ports open for Modbus and for ladder logic uploads to allow the plant personel and legitimate processes to carry out their normal functioning of the ICS. Adversaries who have access to the network can send valid packets with harmful payloads to the PLC through the open network ports. Even if authentication and encryption are being utilized to thwart cyber-attacks against the PLC, an attacker who has compromised a trusted node, such as the HMI, would still be able to launch the attacks studied in this work. Therefore, defensive measures must be put in place by the PLC that can respond to valid packets with harmful payloads.

As seen from EPPIPS's accuracy in predictions and classifying scenarios as either safe versus unsafe, EEPIPS is effective in predicting what a change in a setting would do. Cases in which EPPIPS fails to detect a harmful packet are generally due to situations where the actual scenario becomes barely outside the safety condition and is predicted as safe. Even if a model is fairly accurate to represent an actual physical system, there are cases where minor differences between the model and the actual physical system produce erroneous results. Therefore, it may be appropriate to classify scenarios that are merely barely safe in new category that may be termed 'high risk'.

**Table 9.** Major works for control command analysis-based IDS/IPS.

| Work | Place. | App. | Prev. | Sector | Phys. Sys. Model | Comm. Eval. | Program/ Lad. Logic Eval. | Accuracy | Latency |
|---|---|---|---|---|---|---|---|---|---|
| Chromik | Local | S | Yes | PG | Yes | Yes | No | N/A | N/A |
| Lin | central | S | Yes | PG | Yes | Yes | No | 77.4% - 99.00052% | 5ms – 600ms |
| Etigowni | central | S | Yes | PG | Yes | Yes | Yes | N/A | 150ms |
| McLaughlin | Local | S | Yes | ICS | No | Yes | Yes | N/A | 0.2s – 90s |
| **Werth (EPPIPS)** | Local, embed-ed | S & L | Yes | ICS | Yes | Yes | Yes | 96% - 98% | 30ms – 300ms |

L = learned, S = specification, ICS - Industrial Control System (General), PG = Power Grid.

Also, the results of this work may be compared to other related work. Doing so is not always straightforward because the designs and applications of other IPSs have significant differences even though they are similar in that they make predictions of potential command packets. The latency for Lin's approach ranges from 5ms to about 00 ms based on the size of power grid evaluated [12,13]. Etigowni's approach achieved a maximum latency of 50 ms when evaluating user commands [26]. In the previous work for EPPIPS, latency to determine if a command was unsafe ranged from 0 ms to 00 ms and depended roughly linearly on the number of scan cycles predicted which ranged from 1000 to 10,000 scan cycles. Note that these latency results for EPPIPS assume that EPPIPS is running in an embedded system with the hardware of a Raspberry Pi 3 Model B+ [21]. The accuracy of Lin's approach to detecting commands with the adaptive power flow analysis leading to unsafe conditions is above 99%. However, Lin showed that the accuracy could be as low as 77.4% when using the DC power flow analysis for the 30-bus power system. Such results on accuracy may be more relatable to the results in this work. Chromik, another author whose work is similar, does not provide latency concerning the experiments. Nor does Chromik use accuracy in the evaluation. Chromik instead evaluated a set of case studies and confirms that the approach can detect specific harmful commands for a power system [10,11]. Even though Chromik's and Lin's works both evaluated incoming command packets to predict if they are harmful to a given cyber-physical system, they have fundamental differences to this work. Chromik and Lin focus on power systems where a change in an actuator, such as a breaker, has an almost immediate effect. On the other hand, in the case of EPIPS, the state of the physical system evolved over time more so, and there are many future cycles of the PLC that must be evaluated. Furthermore, EPPIPS not only evaluates the changes to actuators but also the changes to settings in the ladder logic program that ultimately affect the actuators as the ladder logic runs. Therefore, it is difficult to make direct comparisons but it is possible to summarize various features and to compare latency and accuracy in some cases. In addition, McLaughlin developed methods to evaluate commands, or control signals [8], and also ladder logic [9] using a specification- or policy-based approach. His approach involved instrumenting the code to be evaluated and performing symbolic execution to rigorously test the code with a latency ranging from roughly 0.2 second to 0 seconds. In contrast to the other methods described, Mclaughlin's method did not involve a model of a physical system. This model is useful for EPPIPS for making predictions of a physical process whose dynamic behavior is related to how fast or intense an actuator operates. Table 9 summarizes features of the most relevant and comparable IDSs/IPSs of this work. These features include placement (local or embedded in the PLC or central to SCADA master or HMI), the approach (learned or specification-based), whether the IDS/IPS is capable of preventing detected attacks and is therefore an IPS, whether the IPS/IDS uses a model of the physical system, and whether the IDS/IPS evaluates commands or ladder logic in addition to

quantifiable attributes such as accuracy and latency. These are given as a set of ranges that vary on different conditions described previously.

Also, EPIPPS demonstrated through the results that it can detect hidden potential for harm. Commands sent to the PLC to change settings can appear as normal from a network perspective. However, the effects of these commands are unknown unless deep packet inspection of packets that encapsulate the command and also prediction is performed. With the results established in this work, EPPIPS has shown that it can perform the deep packet inspection and make a prediction on whether a given valid command is harmful.

## 7. Conclusions

The results of this work show that EPPIPS was able to classify the vast majority of the simulated scenarios correctly. A summary of the major contributions in this work are as follows: First, the author created multiple testbeds and other representative equipment - (1) Single Station Pipeline with Variable Speed Pump, (2) Storage Tank with Variable Speed Pump, (3) and Safety Instrumented System (SIS). Second, the author developed EPPIPS, which was designed to protect the PLCs against malicious commands and settings that perturb the physical system to unsafe states. Third, there were limited specifications for safety and imminent danger that a domain expert was required to add. Only the safety ranges were necessary to provide since ARX models were trained from time-series data. Fourth, EPPIPS was successful in responding to cyber-attacks by preventing the PLC from processing them in certain cases. Fifth, analysis and evaluation demonstrated that EPPIPS had reasonable accuracy. Sixth, the results of this work demonstrated that the EPPIPS was effective when dealing with cyber-attacks that were designed as valid commands and impacted the physical system to cause unsafe conditions. The methods were especially important for cyber-physical systems since computerized devices that interact with physical systems can impact the physical world. Predicting this impact was important for defending these systems.

Future work will be to develop more sophisticated methods for detecting time bombs and logic bombs in ladder logic where the code of the ladder logic can be analyzed. This analysis would involve systematically examining the paths of execution in the code, their effects on the cyber-physical system and the conditions that trigger these paths. This is necessary to understand if a given path of execution is triggered by some inherent timer or a set of conditions. Therefore, it would be possible to detect if a given piece of code is harmful or not. This would increase the insight that can be gained from a given ladder logic program to a greater extent than what can be understand by merely simulating the code with a model of the physical system alone, which may only be performed for a limited number of scan cycles. Other work may involve extending EPPIPS to protecting multiple PLCs for a given cyber-physical system, and to

test different ways of modeling the physical system. Because of the fundamental differences between EPPIPS and the works by Lin and Chromik, it may be necessary to rely on this future work to directly compare with metrics such as and the many statistics calculated beyond latency and accuracy. This work presents an initial version of EPIPPS that may serve as a baseline for comparison as new similar research and developments occur.

## Acknowledgments

This research was conducted at the University of Alabama in Huntsville in support of Aaron W. Werth's dissertation, which he completed in the fall of 2020. This material is based upon work partially supported by the National Science Foundation under Grant No. 1753900.

## Disclosure statement

No potential conflict of interest was reported by the authors.

## Funding

The work was supported by the National Science Foundation [1753900]

## ORCID

Aaron W. Werth http://orcid.org/0000-0003-1936-5627

## References

[1] Giraldo J, Sarkar E, Cardenas AA, et al. Security and privacy in cyber-physical systems: a survey of surveys. IEEE Des Test. 2017;34(4):7–17.
[2] Langner R. Stuxnet: dissecting a cyberwarfare weapon. IEEE Sec Privacy. 2011 May-June ;9(3):49–51.
[3] Abrams M, Weiss J. Malicious control system cyber security attack case study–maroochy water services, Australia. McLean, VA: The MITRE Corporation; 2008.
[4] Bilge L, Dumitras T. An empirical study of zero-day attacks in the real world. CCS. 2014;12:16–18.
[5] Hahn A, Thomas RK, Lozano I, et al. A multi-layered and kill-chain based security analysis framework for cyber-physical systems. Int J Crit Infrastruct Prot. 2015;11:39–50.
[6] Huang YL, Cárdenas AA, Amin S, et al. Understanding the physical and economic consequences of attacks on control systems. Int J Crit Infrastruct Prot. 2009;2(3):73–83.
[7] McLaughlin S. On dynamic malware payloads aimed at programmable logic controllers. Proceedings of the 6th USENIX conference on Hot topics in security, HotSec'11; Berkeley, CA, USA: USENIX Association; 2011.
[8] McLaughlin S. Cps: stateful policy enforcement for control system device usage. Proceedings Of the 29th Annual Computer Security Applications Conference, ACSAC '13 New Orleans, Louisiana, USA; ACM; 2013. p. 109–118.

[9] McLaughlin SE, Zonouz SA, Pohly DJ, et al. A trusted safety verifier for process controller code. NDSS. 2014;14.
[10] Chromik JJ, Remke A, Haverkort BR. What's under the hood? Improving SCADA security with process awareness. Cyber-Physical Security and Resilience in Smart Grids (CPSR-SG), JointWorkshop on Vienna, Austria; IEEE; 2016. p. 1–6.
[11] Chromik JJ, Remke A, Haverkort BR. Improving SCADA security of a local process with a power grid model. ICS-CSR. 2016;4:114–123.
[12] Lin H, Slagell A, Kalbarczyk Z, et al. Semantic security analysis of SCADA networks to detect malicious control commands in power grids. Proceedings of the first ACM workshop on Smart energy grid security CopenHagen, Denmark; 2013. p. 29–34.
[13] Lin H, Slagell A, Kalbarczyk Z, et al. Runtime semantic security analysis to detect and mitigate control-related attacks in power grids. IEEE Transactions on Smart Grid; 2016.
[14] Werth AW, Morris TH. Intrusion prevention for payloads against cyber-physical systems by predicting potential impacts. J Cyber Secur Technol. 2022;6(3):113–148.
[15] Eckhart M, Ekelhart A. A specification-based state replication approach for digital twins. In Proceedings of the 2018 workshop on cyber-physical systems security and privacy Toronto, Canada; 2018. p. 36–47.
[16] Gehrmann C, Gunnarsson M. A digital twin based industrial automation and control system security architecture. IEEE Trans Ind Inform. 2019;16(1):669–680.
[17] Akbarian F, Tärneberg W, Fitzgerald E, et al. A security framework in digital twins for cloud-based industrial control systems: intrusion detection and mitigation. 2021 26th IEEE International Conference on Emerging Technologies and Factory Automation (ETFA) Vasteras, Sweden; IEEE; 2021. p. 01–08.
[18] Varghese SA, Ghadim AD, Balador A, et al. Digital twin-based intrusion detection for industrial control systems. In 2022 IEEE International Conference on Pervasive Computing and Communications Workshops and other Affiliated Events (PerCom Workshops) Pisa, Italia; IEEE; 2022 March. p. 611–617.
[19] Francia G, Hall G. Digital twins for industrial control systems security. In 2021 International Conference on Computational Science and Computational Intelligence (CSCI) Las Vegas, Navada, USA; IEEE; 2021 Dec. p. 801–805.
[20] Govil N, Agrawal A, Ole Tippenhauer N. On ladder logic bombs in industrial control systems. In: katsikas, sokratis, Cuppens, Frederic, Cuppens, Nora, Lambrinoudakis, Costas, Kalloniatis, Christos, Mylopoulos, John, Anton, Annie, Gritzalis, Stefanos, editors. Computer security. Cham: Springer; 2017. p. 110–126.
[21] Werth AW, Morris TH. Intrusion prevention for payloads against cyber-physical systems by predicting potential impacts. J Cyber Secur Technol. 2022 6 3 ;1–36.
[22] Aaron W, Morris TH. A specification-based intrusion prevention system for malicious payloads Choo, Kim-Kwang Raymond, Peterson, Gilbert L. eds. . In: National cyber summit. Cham: Springer; 2019. p. 153–168.
[23] What is a safety instrumented system? RealPars. [2019 Mar 31; cited 2019 May 8]. [Online]. Available from: https://realpars.com/safety-instrumented-system/#
[24] Di P, Alessandro YD, Carcano A. TRITON: the first ICS cyber attack on safety instrument systems. Proc Black Hat USA. 2018;2018:1–26.
[25] Alves T, Das R, Werth A, et al. Virtualization of SCADA testbeds for cybersecurity research: a modular approach. Comput Sec. 2018;77:531–546.
[26] Etigowni S, Jing Tian D, Hernandez G, et al. CPAC: securing critical infrastructure with cyber-physical access control. In Proceedings of the 32nd Annual Conference on Computer Security Applications Los Angelas, California, USA; ACM; 2016. p. 139–152.