VIRTUALIZATION-BASED RESILIENCE APPROACHES FOR INDUSTRIAL CONTROL SYSTEMS

by
James D. Cervini

A dissertation submitted to Johns Hopkins University in conformity with the requirements for the degree of Doctor of Engineering

Baltimore, Maryland

April 2022

Abstract

Industrial Control Systems (ICS) and their components perform cyber-physical functions. In the context of critical infrastructure, these functions are vital to modern life. Programmable Logic Controllers (PLCs) are prominently found in ICS environments and execute the operational logic of the system. The continued escalation of cyberattacks targeting ICS and their PLCs serves as motivation for increasing system resilience. This dissertation analyzes domain cyber-threats and demonstrates novel approaches which utilize virtualization, containerization, input/output multiplexing, cryptographic attestation, software defined networking, security orchestration, and PLC runtimes to advance PLC trust and resilience while facilitating integration into past, present, and future systems. The research approaches were proven using physical ICS testbed environments with experimentation results showcasing enhanced control system trust and resilience.

Primary Reader and Advisor: Aviel Rubin

Secondary Reader(s): Lanier Watkins and Anton Dahbura

Acknowledgments

I would like to thank the people whose constant support empowered me to pursue my life-long dream of doctoral education. I am beyond grateful to my advisor, Professor Aviel Rubin, for his guidance and insight as I charted new ground on the road towards the Doctor of Engineering (DENG) degree. Avi's knowledge and experience proved invaluable and ensured I was equipped with the tools for success while providing the opportunity to explore impactful research topics.

I would also like to thank Dr. Lanier Watkins for serving on my supervisory committee and his support as I navigated the academic process. Lanier's continuous and tactical advice influenced how I published and presented my research. I am also very grateful to Dr. Anton Dahbura for serving on my supervisory committee and providing beneficial input throughout this program. I would also like to thank Dr. Yinzhi Cao and Dr. Susan Hohenberger Waters for serving on my research proposal examination board.

This pursuit of my doctoral education was made possible by The Johns Hopkins University Applied Physics Laboratory (JHU/APL). I am thankful for the opportunities, facilities, support, and resources provided to me throughout this process. I would like to highlight and thank Tao Jen for orchestrating organizational support and providing the leading-edge laboratory environment where my research occurred.

The research included in this dissertation was published at various academic venues. I would like to thank the co-authors for their contributions. They include Avi Rubin [1] [2] [3], Lanier Watkins [1] [2] [3], Daniel Muller [3], Alexander Beall [3], and Joseph Maurio [3]. Expanded variations of these publications are contained within this dissertation.

I would like to thank my fiancée and soon-to-be wife Gabriela, whose unending support and encouragement inspires me to set and achieve ambitious goals. Lastly, I would like to thank my mother Carol Cervini-Wong, whose prioritization of my education led me down a path which brought me to this program and instilled my value of education which served as a motivator to pursue this degree.

Dedication

This thesis is dedicated to my fiancée and future wife Gabriela, my mother Carol Cervini-Wong, and my Grandmother Peggy Metty for their immeasurable and ceaseless love, support, and inspiration throughout this journey.

Table of Contents

List of Tables

List of Figures

1 Introduction

This dissertation explores and analyzes the challenges impacting the cybersecurity of industrial control systems (ICS) and exhibits novel research approaches for improving their cyber-resilience. Impacting the ICS domain involves considering the unique domain challenges, many of which stem from their societal importance. This body of research conforms to these domain constraints while establishing a new paradigm of ICS resilience.

This body of work explores the current and future states of ICS cyber threats and utilizes this examination to guide and benchmark subsequent research. We researched approaches which generate and leverage high-fidelity hardware abstraction to enable a self-healing ICS continuity when faced with cyber compromise. This peer-reviewed work addresses critical domain challenges and applies resilience to many systems with unmatched societal significance while serving as an enabler for future research for years to come.

1.1 Industrial Control System Concepts and Definition

ICS have a global presence and are responsible for the logical operation of physical industrial processes. Their societal importance stems from their prevalence in critical infrastructure, with common representation in the energy, water treatment, chemical, manufacturing, oil, and natural gas sectors. The United States Department of Homeland Security (DHS) Cybersecurity and Infrastructure Security Agency (CISA) asserts that destructive attacks targeting these systems would have a monumental impact on public safety, health, and security [4].

ICS are fundamentally reliant upon programmable logic controllers (PLCs) for their operations. PLCs are ruggedized, embedded, real-time computers which cyclically execute programmed logic. This logic is the driving force behind control system automation and is

stimulated by operator input and the ingestion of data from physical sensors. PLCs can interact with physical control processes by generating electrical control signals. Examples of common control signals include 0-10 volts (V) and 4-20 milliampere (mA). These logic-driven signals control the actuator components which comprise the physical process. Most industrial PLCs use a slot-based architecture where different slots can be occupied by modules selected to meet the processing, networking, cabling, and control signal requirements of an ICS. Figure 1-1 below shows an Allen Bradley ControlLogix PLC, which holds a large market share in North America and is utilized as a benchmark throughout experimentation [51].

Figure 1-1 An Allen Bradley PLC Chassis Wired to a Control System [9]

Operators interface with control systems through various means. Typically, operators utilize human machine interfaces (HMI). HMIs can be deployed as software which is installed and executed on commodity information technology (IT) hardware. Alternatively, many HMIs are installed as ruggedized embedded computing touchscreen panels located in close proximity to the control process as seen in Figure 1-2. This hardware-based installation provides a mechanism for localized control. Lastly, operators can manually manipulate the control process. Manual controls, such as physically turning a valve, forgoes the control system's automation and relies

on the operator's understanding of the system for operation. Manual controls lack the efficiency and precision of the automated controls and as a result, are typically reserved for emergency or maintenance operations.

Figure 1-2 An Allen Bradley Touchscreen HMI [10]

1.2 Resilience Concerns of Industrial Control Systems

Initially, ICS environments exclusively utilized manual controls for operations, eventually transitioning to electromechanical and pneumatic logic. At this time, the biggest threat to process continuity was mechanical failure, physical security, and malicious insiders. In the pursuit of cost-

savings and enhanced capabilities, PLCs were conceptualized as a way to automate process functions and increase efficiency. Likewise, modern HMIs were developed as a means to digitize expensive indicators and tactile interfaces. As this efficiency and cost savings pursuit continued into the modern era, HMIs, PLCs, and ICS increased in complexity, introducing additional threats to process continuity.

Due to the criticality of many ICS processes, minimal downtime is allotted for system upgrades. Consequently, this harsh uptime requirement combined with high component cost results in long PLC replacement lifecycles and establishing a culture of not replacing functional hardware, culminating in rampant use of insecure and end of life components. Additionally, modern interconnectivity enabled these frequently vulnerable systems to be accessed remotely through the use of supervisory control and data acquisition (SCADA) systems. This centralized management and remote access facilitates the monitoring, maintenance, and operation of critical infrastructure, all-the-while exposing the systems to a larger number of potential cyber threats. Additionally, the remote locality of these SCADA assets results in less effective manual controls during emergencies as operators need to physically travel to the target location, increasing response times. Current best practices emphasize a castle wall approach to ICS cybersecurity. As the name implies, a castle wall approach emphasizes the implementation of strict cyber-perimeters. However, as the size and prevalence of SCADA networks grow so must their castle walls, and this larger footprint increases the likelihood for implementation error. As history has shown, once castle walls are breached, battling an adversary becomes incredibly difficult. This also applies to ICS networks due to the vulnerable disposition of the connected devices, devices which perform many critical societal functions.

Organizations which own and operate critical infrastructure also require IT components and infrastructure to support their operations. Services like billing, payment processing, e-mail, and employee timekeeping are logistical enablers of an organization's effectiveness. It is understood that segmenting the control network from the IT network is imperative, yet many ICS compromises originate from the IT infrastructure and pivot to the control network. This often occurs due to misconfiguration or lack of cybersecurity awareness with thousands of industrial controllers observable from the Internet. Figure 1-3 illustrates a typical ICS architecture.

These systems and their processes are lucrative targets for malicious adversaries due to their criticality and vulnerability. Many ICS environments have been the subject of targeting by malicious actors with some recent prominent examples at the time of writing being the Colonial Pipeline and the Oldsmar Water Treatment Plant [8] [2].

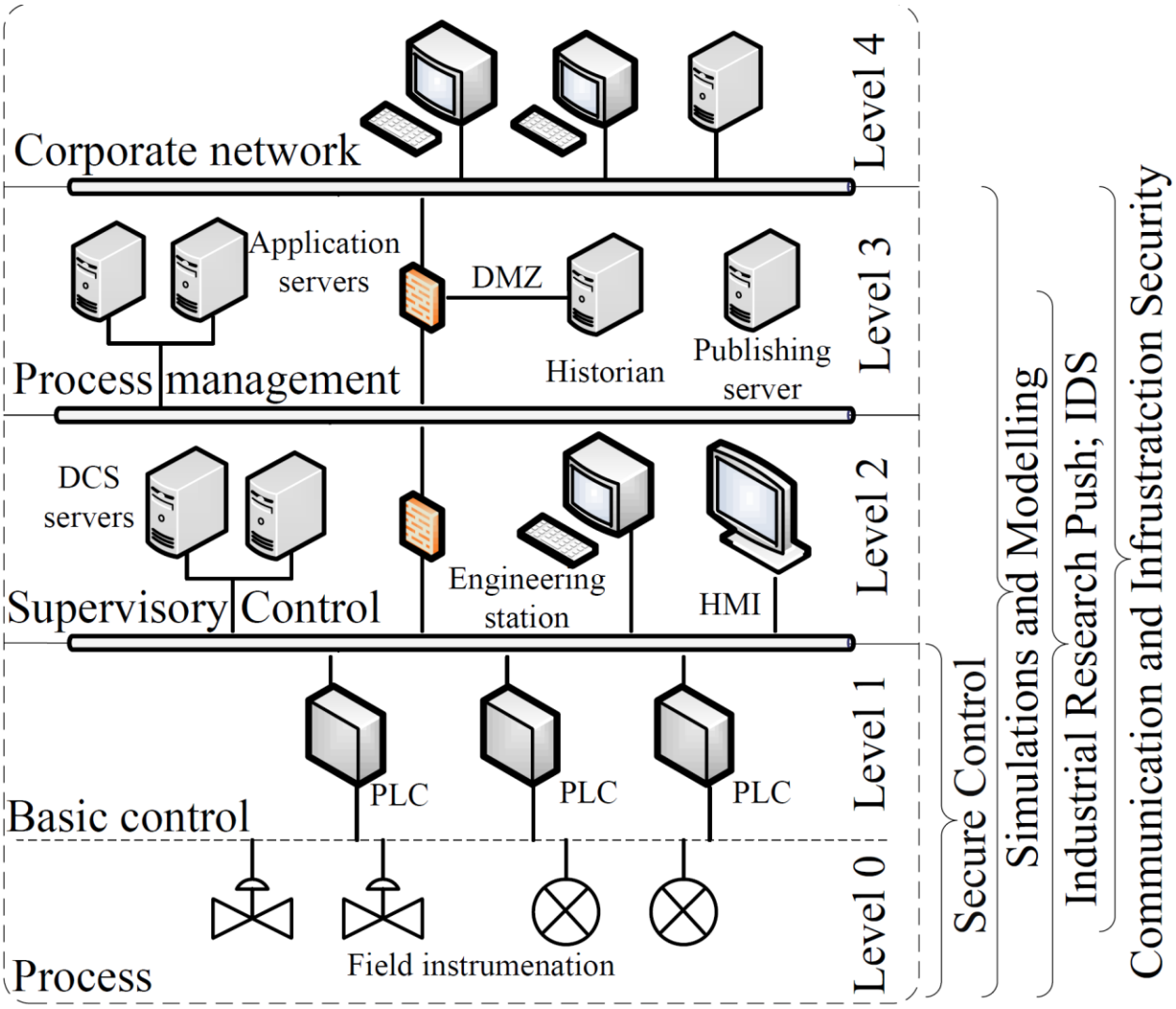


Figure 1-3 An ICS Architecture Mapped to Security Solution Types [7]

A critical challenge affecting ICS resilience is the inability to respond and recover from cyber incidents. Generally, the approach the ICS domain utilizes to respond to faulty hardware is the same response applied to cyber compromises. This is ineffective versus the plentiful and diverse outcomes associated with cyber compromises. Assuming operators have sufficient situational awareness and can correctly discern the incident resulted from a cyberattack and not faulty hardware, current troubleshooting procedures often consist of:

- Physically locating the affected devices, a challenge in large and sometimes hazardous industrial environments.
- Attempting to use manual controls and safe shutdown procedures to enter a safe state.
- Power-cycling the PLCs if the physical process allows.
- Manually reflashing the PLCs' logic and/or firmware.
- If all else fails, physically replace the devices.

Ultimately, these procedures are ineffective versus a competent cyber adversary utilizing basic persistence techniques. This research answers this critical challenge of ICS cyber resilience through researching approaches which empower these systems and processes to effectively respond and recover from cyber threats.

1.3 Vision and Approach

The National Institute of Standards and Technology (NIST) established a cybersecurity framework which defines effective measures for ensuring an effective cybersecurity posture. The framework consists of the following five functions: identification of assets, protection of assets, detection of anomalous or malicious behavior, response to the detected indicators, and recovery of assets [5]. These functions have applicability to the ICS domain with this body of work focusing on the introduction of novel approaches to the response and recovery functions. Inspired by the resiliency benefits hardware abstraction enables in the IT domain, this body of research envisions a future in which these same benefits are realized in the operational technology (OT) domain.

The ICS domain's culture of efficiency and cost savings are typically at odds with proactive cybersecurity investments. Understanding this domain culture, the research in this dissertation aims to enhance the resiliency of control systems while also providing cost savings and efficiency benefits in an effort to increase the likelihood of approach adoption.

This research vision is achieved through multiple research thrusts culminating in an approach which leverages hardware abstraction to achieve automated process recovery when subjected to various cyber compromises while adhering to domain constraints. Initially, the research focused on understanding the current and future ICS threat spaces and using this understanding to define what an effective approach should be capable of mitigating. Subsequently, a containerization-based approach was researched and demonstrated to prove approach feasibility and mitigation potential. Lastly, a virtualization-based approach expands on the containerization-based approach by defining an architecture which incorporates cryptographically assured trust, low-effort integration, and novel automation.

1.3.1 Researching the Threat Landscape [2]

Understanding the cyber threats impacting the ICS domain is critical to developing an effective and applicable mitigation approach. Additionally, understanding the realized threats enables the extrapolation of future threats to ensure research maintains maximum long-term applicability while highlighting the risks associated with domain inaction. Therefore, this dissertation explores a recent and relevant example of a cyberattack targeting critical infrastructure. Through attack dissection, attack characteristics are represented as variables used to define the future theoretical threats and showcase their potential impact. We use this analysis

to outline easily adoptable partial solutions while defining an effective mitigation approach which ensures process continuity.

1.3.2 Hardware Abstraction [1] [3]

A fundamental cornerstone to this body of work is the exploration of containerization and virtualization-based hardware abstraction to enhance ICS components with a previously unseen level of flexibility. This flexibility, when stimulated with indicators and decision-making enables ICS processes to recover from malicious activity and ensure continuity. Empowered by modern computing and networking performance, we research, realize, and demonstrate the resiliency benefits and enhancements made possible by this approach. Experimentation occurred on water treatment, water chilling, and power distribution environments.

1.3.3 Cryptographic Trust [3]

Relying on an underlying approach or technology requires trust. Therefore, methods of assuring the trustworthiness of this research approach are paramount to enhancing the approach and facilitating adoption. This research utilizes hardware-based cryptographic attestation to verify the internal configuration of our solution reflects expected values. This data is fed to a decision-making engine that enables additional logical relevance to the approach's flexibility benefits. This approach's effectiveness and applicability was demonstrated through experimentation in a power distribution environment.

1.3.4 Automation and Orchestration [1] [3]

To best utilize the flexibility provided by the approach's hardware abstraction, we enhanced the approach through the novel incorporation of automation and orchestration. Indicator

ingestion, data coordination, and automated response execution was introduced to the approach's architecture, incorporating a decision-making element to the novel action space realized by the approach's flexibility. This automation and orchestration enhanced approach was explored, and its effectiveness was demonstrated through experimentation in a power distribution environment.

1.3.5 Approach Adoptability [1] [3]

A key consideration throughout the research was the approach's adoptability. Ease of adoption is important due to the infrequent upgrade cycles previously discussed. Approaches which are cumbersome to integrate, fail to adhere to domain constraints, or are inconsiderate of domain culture are unlikely to see adoption, and subsequently, realized impact. In contrast, research which overemphasises these considerations can find their research solutions constrained. Therefore, this research strikes a balance by defining a novel approach which simultaneously serves as an emerging architecture for forthcoming resilient control systems while being easily adopted to existing systems. This assurance of easy adoptability is achieved through the research's integration and testing within experimentation environments consisting of commonplace commodity ICS devices.

1.4 Outline of this Work

Each chapter in this dissertation explores an aspect of the ICS resilience problem and details our analysis of related works, proposed solutions, experimentation processes, evaluation criteria, and results.

- Chapter 2 examines the present and future ICS threat landscape and defines the criteria for a successful solution.
- Chapter 3 explores an approach which utilizes hardware abstraction to enhance the resilience of ICS.
- Chapter 4 explores an approach which establishes an automated ICS continuity architecture utilizing trusted computing.
- Chapter 5 summarizes the previously detailed work and conveys its lasting impact.
- Chapter 6 lists the references cited in this body of research.

2 Don't Drink the Cyber: Extrapolating the Possibilities of the Oldsmar Water Treatment Cyberattack

2.1 Introduction

The water and wastewater systems sector is labeled as a vital critical infrastructure sector by the United States Department of Homeland Security, describing this sector as "essential to modern life" [18]. Water treatment responsibilities are commonly performed by municipalities with meager budgets for cybersecurity resources. According to a report which surveyed water sector utilities, 44.8 percent claimed less than one percent of their overall budget was dedicated to OT cybersecurity [26]. The high impact of water treatment disruption and malicious manipulation when paired with an underfunded defensive cyber-posture results in an emerging cyber warfare paradigm which poses a direct threat to the quality of life for millions of people. The 2021 water treatment cyberattack orchestrated in Oldsmar, Florida represents a relevant example of cybersecurity shortcomings prevalent in underfunded municipalities. While Oldsmar is home to approximately 15,000 residents, coordinated cyber warfare campaigns targeting these vulnerable critical systems represent asymmetric threats which could impact millions.

This work contributes to the domain by researching a relevant OT municipal cyberattack with unrealized impact while framing it in the context of cyber warfare at scale. Furthermore, this research illustrates how an increased likelihood for impact realization is possible while outlining mitigations mindful to the operational constraints of municipalities. Section 2.2 of this chapter will describe and dissect this attack, with section 2.3 ultimately showing how slight modifications to the attacker's tactics, techniques, and procedures (TTPs) could have resulted in a devastating cyberattack. Subsequently, section 2.4 describes low-cost mitigations which aim to mitigate the

exhibited threats with various mitigation types described in each sub-section. Section 2.5 outlines an effective research approach which further enhances system resilience. Lastly, section 2.6 concludes the chapter with closing thoughts.

2.2 The Oldsmar Water Treatment Attack

The attack began on February 5th, 2021, at 0800 Eastern Standard Time (EST) where the attacker leveraged compromised credentials and the common remote access software TeamViewer to login to a plant operator's console. Following the login, the attacker immediately logged off and disconnected the session [23]. This was likely the attacker confirming system, connection, and credential validity. At this point, the attacker could have also taken a screenshot of the console supporting the planning of further actions. The console was manned by an operator at this time and the anomalous behavior was mentally noted, but no action was taken. Later that day, at 1330 EST the attacker logged back into the operator console and used the plant's HMI to command the increase of sodium hydroxide, also known as lye, to water which supplied thousands of people. Lye exposure can result in painful burns and permanent damage if ingested. The lye was increased from 100 parts per million (ppm) to 11100 ppm, an increase of over 100 times the intended amount. The operator observed this anomalous behavior and commanded levels back to normal values. The operator then notified a plant supervisor, and the TeamViewer access was removed. This attack also occurred in the days leading up to the Super Bowl on February 7th which was being hosted 12 miles away from the facility, likely drawing more people to the area. According to county officials, it would have taken 24 to 36 hours of an increased mixture setting to have the malicious chemical mixture reach a point of distribution where it could have been consumed by customers. Additionally, county officials cited the use of

potential hydrogen (pH) level sensors with alarm reporting functionalities which would have notified operators of an imbalance [23].

Figure 2-1 Oldsmar's Bruce T. Haddock Water Treatment Plant [11]

This attack lacks much of the complexity seen in previous cyberattacks targeting OT. Current reporting suggests the attacker did not attempt to evade detection and the attacker did not deploy custom scripts or programs, instead opting to leverage the existing operator interface to command their effect. While some would see this as an indicator of a low-skilled adversary, it is often the case that an attacker, regardless of skill, will use the easiest and fastest means to achieve a desired outcome. This can result in highly skilled attackers using techniques seen as simplistic or effortless. Therefore, it can be difficult to assess the attacker's skill in this case with the only potential indicator being the assumption that they did not meet their desired effect. Table 2-1 below breaks down relevant characteristics of the water treatment attack that occurred on 02/05/2021.

Table 2-1 The Characteristics of the Oldsmar Water Treatment Cyberattack

Attack Characteristic	**Oldsmar Water Treatment Attacker**
Attack Date	02/05/2021

Attack Time	1330 EST
Access	TeamViewer Remote Service
Attack Execution	Process HMI
Detection Evasion	None Reported
Persistence	None Reported
Response Function Obstruction	None Reported
Operational Impact	▪ Momentary Control Manipulation ▪ Reduction of System Confidence

This table incorporates the current understanding of the attack's attributes and maps them to characteristics of which some are derived from the MITRE ATT&CK for ICS knowledge base [12].

2.3 Modification of Attacker Tactics Techniques and Procedures

Using the 02/05/2021 attack as a baseline and modifying certain attack characteristics generates two new theoretical attacker models. These two new attacker models vary by operational preferences for stealth and persistence but share the same mission of impacting the delivery of potable water, resulting in either an inability to deliver potable water or the delivery of dangerous water. Attack success is defined by the ability to command the increased addition of lye for the 24-to-36-hour timeline provided by county officials and the ability to degrade or eliminate the situational awareness provided by the system's pH sensors [23]. Given its proven success, all three attacker models will share a commonality of access via the TeamViewer remote service. A high-impact differentiator between the theoretical attacks and the 02/05/2021 attack is the date and time of attack execution. This temporal differentiator requires no additional skills

to utilize and results in a delayed response and higher chance for mission success. The Oldsmar water treatment plant's normal hours of operation are publicly available via open-source information [17]. Leveraging this information to launch attacks outside the normal hours of operation, both theoretical attackers will execute their operations just eleven hours and thirty minutes later compared to the 02/05/2021 attack, opting for an attack on 02/06/2021 at 0100 EST. Executing the attack at this time reduces the likelihood of an operator noticing the remote access and could increase the response time for lye level correction. For the sake of realism, neither theoretical attacker will leverage supply chain compromises, zero-day exploits, or low-level firmware manipulation as these concepts require a substantial amount of time and resources to employ. Table 2-2 below displays the various attacks and their TTPs.

Table 2-2 A Comparison of Attacker Model Characteristics

Attack Characteristic	**Oldsmar Water Treatment Attacker**	**Theoretical Attacker A**	**Theoretical Attacker B**
Attack Date	02/05/2021	02/06/2021	02/06/2021
Attack Time	1330 EST	0100 EST	0100 EST
Access	TeamViewer Remote Service	TeamViewer Remote Service	TeamViewer Remote Service
Attack Execution	Process HMI	Process HMI	Malicious Background Process
Detection Evasion	None Reported	Ransomware	System State Spoofing
Persistence	None Reported	None	Execution on Start-Up
Response Function Obstruction	None Reported	Ransomware	▪ Command Blocking ▪ Alarm Suppression
Operational Impact	▪ Momentary Control Manipulation ▪ Reduction of System Confidence	▪ Control Manipulation ▪ Reduction of System Confidence ▪ Data Destruction	▪ Lengthy Control Manipulation ▪ Delivery of Unsafe Water

		▪ Lengthy Inability to Deliver Potable water ▪ Possible Delivery of Unsafe Water	

2.3.1 Theoretical Attacker A

Theoretical attacker A is impartial to attack discovery, instead opting to quickly impact the system while creating a difficult path for system recovery. Following the 0100 EST command to increase the lye amounts in the water supply, theoretical attacker A launches a ransomware attack on the system's computers to include operator stations. This ransomware attack reduces the operator's ability to control the system and can stifle the alerting of the pH imbalance due to a lack of situational awareness provided by their system interfaces. Communication could also be impeded if ransomware spreads to e-mail servers. A ransom note requesting bitcoin in exchange for decryption would manipulate responders into thinking the attack was financially motivated and concealing the attack's true intentions. Ransomware also has the added impact of encrypting log files, rendering them useless and increasing the difficulty of attack attribution.

According to a recent study, the number of reported ransomware attacks continues to increase yearly, one of which was the colonial pipeline ransomware attack [24]. This example illuminates the reality that malicious actors are targeting critical infrastructure systems which provide key services. This relevant example and continuing trend of ransomware use supports the realism of this theoretical attack scenario. Another emerging trend in ransomware is what is known as double-extortion ransomware, where attackers will apply additional pressure by threatening the release of sensitive stolen data in addition to system encryption [20]. To this end, it is possible that threats of malicious physical process manipulation carried out by dormant logic

bombs will accompany future ransomware attacks on critical infrastructure to add additional pressure to the ransom. To illustrate the timelines associated with ransomware, a ransomware attack on the city of Baltimore, Maryland took over a month to recover [21]. Acknowledging the differences in scale between the two systems, a report polling 2690 IT professionals claimed that 66 percent did not believe their organization could recover from an unpaid ransomware attack in less than five days [27]. Therefore, it is generally unlikely that full system functionality would be restored in the 24-to-36-hour timeline. Additionally, the odd hours of the attack delays initial attack discovery and subsequently the recovery process. Lastly, the unseen lye command and potential degradation of the system's pH sensing due to encrypted operator interfaces enables the potential delivery of unsafe water. In the event plant personnel discover the lye manipulation but are unable to access their encrypted interfaces to assert positive control, a possible response could be to halt the delivery of water until systems are reconstituted or operate the plant completely manually, sacrificing precision and increasing operating costs. These factors apply additional ransom pressure and lead to a potential reduction in product throughput and quality. In both scenarios, the attacker has achieved their mission and impacted the quality of life for thousands.

2.3.2 Theoretical Attacker B

Theoretical attack B requires additional planning and domain understanding to execute but ensures guaranteed mission success. Attacker B relies on stealth to ensure that operators are unaware of the true state of their systems until dangerous water reaches customers. This attack requires a moderate understanding of cyber-physical systems and cyberattacks. This attacker gains initial system access via the TeamViewer remote access application. Instead of continuing

to use TeamViewer, attacker B establishes a malicious background process which provides subtle backdoor access unobservable to an operator under normal operating conditions. This process is configured to run at start-up, granting it persistence upon reboot cycles. This gives the attacker a long-term foothold to gain additional information on the system's processes, devices, and configuration. With this newfound system understanding, the attacker chooses to execute a man-in-the-middle (MITM) attack. This attack commands the system to add dangerous levels of lye to the water while reporting normal values to the operator. This can be accomplished in multiple ways and the attacker chooses whichever presents the highest chance of success based on initial reconnaissance.

In this theoretical scenario, the attacker chooses to modify the PLC's ladder logic responsible for lye addition and pH reporting. PLCs are ruggedized real-time controllers prominent in many ICS environments which execute pre-programmed logic and interface with physical systems via actuator and sensor signals. The attacker reprograms the device, hard coding a dangerously high value of lye to be added which would otherwise be controlled by the operator or the device's original logic. The attacker also modifies the scaling of any reported values related to lye introduction and pH sensing to reflect normal values. This strips away awareness of the system's true status provided by the physical sensors connected to the PLC. An additional benefit that stealth provides, is the inability to discern cyberattack from system failure. In the event plant personnel were able to detect the pH imbalance through an out-of-band channel, they would more likely attribute the anomaly to a faulty sensor or hardware failure than a cyberattack. As a result, any hardware replacements would prove ineffective considering the attacker's foothold on the system. This attack uses long-term access and stealth to ensure the lye value's change

lasts at least the 24 to 36 hours specified by county officials. It also considers the system's pH sensing and circumvents this via a MITM attack. Even if these operational requirements were not outright stated, the attacker's long-term foothold would have provided sufficient time to understand the system process and form attack requirements. Given this attacker's stealthy approach, it is highly likely that the first indications operators would receive of this attack would be calls associated with consumption of dangerous water, resulting in a successful mission outcome for the attacker.

2.4 Theoretical Application of Potential Low-Cost Mitigations

The following subsections outline various solutions which attempt to mitigate the threats shown throughout this chapter. These solutions prioritize threat mitigation while adhering to the budgetary and personnel constraints limiting many municipal utilities. Even so, all these solutions are directly dependent on the understanding of the current system and a lack of detailed system and configuration knowledge could reduce the impact of these mitigations. Additionally, any misconfiguration of these capabilities could result in a less effective mitigation solution. Lastly, these solutions are best utilized in combination with policies, procedures, and training that outline capability use. Table 2-3 below plots these proposed low-cost mitigations against the characteristics of the three attackers to understand their required effort and effectiveness. For each proposed mitigation, two numbers are provided. The first number represents the mitigation's implementation effort with '1' signifying low effort required for implementation, '2' signifying medium implementation effort, and '3' signifying high implementation effort required. The second number represents the mitigation's effectiveness to the associated attack characteristic with '1' signifying low effectiveness, '2' signifying medium effectiveness, and '3'

signifying high mitigation effectiveness. For example, a solution which is easy to mitigate and highly effective would be given the values 1|3. Mitigations that are irrelevant to an attack characteristic will be denoted as not applicable (N/A).

Table 2-3 Effort and Effectiveness of the Outlined Mitigations

Attack Characteristic \ **Proposed Mitigation**		Remote Access Limitations	Anomaly Detection	Anomaly Prevention	Security Orchestration	Ransomware Mitigation	Training & Awareness
Attack Date & Time	2/5/2021 1330 EST	1|3	1|3	1|3	2|3	N/A	1|3
	2/6/2021 0100 EST	1|3	2|3	1|3	2|3	N/A	1|2
Access	TeamViewer Remote Service	1|3	1|3	N/A	N/A	N/A	1|3
Attack Execution	Process HMI	N/A	3|3	3|3	N/A	N/A	N/A
	Malicious Background Process	N/A	2|3	3|3	N/A	N/A	N/A
Detection Evasion	Ransomware	N/A	1|1	1|3	2|3	1|3	1|3
	System State Spoofing	N/A	3|3	N/A	N/A	N/A	N/A
Persistence	Execution on Start-Up	N/A	2|3	2|3	N/A	N/A	N/A
Response Function Obstruction	Ransomware	N/A	1|1	1|3	2|3	1|3	1|3
	Command Blocking	N/A	3|3	3|3	N/A	N/A	N/A
	Alarm Suppression	N/A	N/A	3|3	N/A	N/A	2|3

2.4.1 Remote Access Limitations

The first mitigation is to establish and ensure a strong cyber-perimeter, this includes restricting and eliminating remote access wherever possible within operational parameters. Ensure devices are patched, prioritizing any internet facing devices. If remote access to plant data

is required for situational awareness but positive control can be limited to on-site personnel, plants should consider implementing a one-way data-diode. If remote access is required for periodic remote maintenance, leave properly configured remote access systems offline, only bringing them online during coordinated maintenance time-windows to minimize internet exposure. Ensuring proper configuration and patching systems does not incur costs associated with new hardware, instead requiring minimal personnel time making it a cost-effective strategy to mitigate potential threats. One-way data diodes do incur hardware costs but are generally available for less than $10,000 and significantly reduce risk if they are the only operationally relevant solution for remote access removal at a given site. Proper remote access limitations would have mitigated the 02/05/2021 attacker and both theoretical attackers. Even so, while ensuring a strong cyber-perimeter is the first line of defense towards threat mitigation, it must be assumed that an attacker will still be able to gain access and that mitigation measures within the cyber-perimeter should be implemented. The following sections outline solutions which apply to internal systems and aim to mitigate threats within these boundaries.

2.4.2 Anomaly Detection and Prevention

A heightened awareness of the communications traversing the internal network can illuminate malicious actors on a network, aid in attack response, motive discovery, and attribution while assisting in the differentiation between hardware failure and cyberattack. While the 02/05/2021 attack was blatantly observed, the two theoretical attacks are executed such that potential observation is minimized. In this case, properly configured network security monitoring and logging would have contained the artifacts and alerting necessary to begin responding to the attack. However, the effectiveness of this mitigation relies on how

operationally relevant its detection mechanisms are as anomaly detection only designed for traditional IT system threats lack much of the domain-specific context present in OT systems. For example, an anomaly detection system comprised entirely of IT threat signatures would be unable to discern the lye increase as malicious. Therefore, a low-cost OT-relevant solution is required to maximize situational awareness in the event of a cyberattack.

Security Onion and ROCK NSM are two popular and free open-source solutions to network monitoring [13] [25]. While these tools are IT-centric, additional rules can be added and customized to provide more OT-relevance. Digital Bond's Quickdraw is an example of free open-source rulesets which can be added to alert on OT threats [22]. Additionally, Malcolm is a free open-source solution which has additional support for the communication types prevalent within ICS environments [15]. The most effective anomaly detection stems from rulesets generated by an understanding of process knowledge. For example, if the command to increase the lye amount should never exceed a certain value, a specific rule can be written to alert on this occurrence. These customized operational-specific rules require a deep understanding of the OT system and process paired with an understanding of the IT infrastructure required to implement them. This network monitoring solution requires minimal hardware investment, only a host machine to run the software and a configuration of network infrastructure to support port mirroring which provides data for ingestion. However, this solution does require a significant amount of personnel time to configure, install, and monitor. OT-tailored solutions which utilize artificial intelligence to automatically baseline a system and alert on deviations of standard behavior exist, but they require a significant capital investment compared to the free open-source solutions but require far less personnel investment. Therefore, a solution choice should be determined based on a

site's available resources. However, regardless of the solution, policies and procedures which outline actions following alert generation should be implemented to best utilize the capability.

In the event theoretical attack A is successful, and operators arrive to find machines presenting ransom notes requesting payment for system decryption, examination of the anomaly detection artifacts highlights indicators of when and how the attack occurred. If the anomaly detection has enough OT specificity, responders would be alerted of the lye change and the attacker's true motivations. If the alerting is configured to be sent to offsite personnel via e-mail or short message service (SMS) transmission the attack response becomes timelier, minimizing the impact of both theoretical attacks' off-hour execution. Lastly, due to the anomaly detection system's ingestion of mirrored traffic, it can be segmented from the network such that a ransomware attack would be unable to modify its artifacts.

Theoretical attacker B's stealth makes detection more difficult, but not impossible. Up-to-date and customized OT rulesets would detect the reprogramming of the PLC allowing plant personnel to attribute system abnormalities to a cyber incident. Additionally, the associated alerts would attribute the attack to a source device, allowing attack responders to segment and investigate the system with the malicious background process.

2.4.2.1 Bump-In-The-Wire Solutions

Anomaly detection and prevention can be backfitted into existing systems using bump-in-the-wire (BITW) solutions. These solutions are implemented using hardware which sits on the network in-between devices. They monitor each communication packet traversing the network, alerting and blocking traffic which trigger pre-programmed security rulesets. Unlike the network monitoring solutions showcased in section 4.2 which used port mirroring to ingest data, the BITW

solutions are positioned in the network to be able to act when a rule is triggered as opposed to just alerting an operator. These solutions require an investment in hardware to place throughout the system. The size of an installation will determine the significance of the hardware investment. However, if financial resources are limited, the installation of BITW solutions can be prioritized to critical systems.

In theoretical attack A, the command to increase the lye amount could be detected and automatically dropped. Accompanying alerts would give responders a timeline of events and provide insight into the attacker's true intentions. A similar ruleset prohibiting device reprogramming unless expressly allowed through operator coordination would eliminate theoretical attacker B's MITM attack, alerting reduces the adversary's stealth advantages, and provide responders with awareness to support further remediation.

2.4.3 Security Orchestration

Security orchestration automates many of the actions a network defender would take in combating a cyberattack via pre-defined playbooks. Orchestration could be a huge benefit to installations with little to no personnel dedicated to actively defending the network. Free orchestration solutions allow for a pre-defined limit of automated actions to occur within a time window, but these constraints would likely be within most requirements of small municipal control systems. Orchestration itself requires a minimal investment of capital and personnel resources, but it does require a system with security solutions implemented to be most effective. For example, orchestration could automate the generation and implementation of a new firewall ruleset based on observed malicious behavior. This scenario would require a competent network monitoring solution to observe and alert on the malicious behavior and would also require a

firewall compatible with your orchestrator to ingest the new ruleset. If the orchestrator does not have sufficient data to inform its decision-making or the proper mechanisms to enact impactful defenses its effectiveness is suppressed.

The ability to automate the response and adaptation to cyber threats in real-time is a valuable asset and can negate the benefits of both theoretical attacks' off-hour executions. Additionally, the automated orchestration empowers underfunded teams to achieve more despite limited resources. Assuming a compatible security architecture is in place, the automation of the response process helps to ensure recovery occurs within the 24-to-36-hour timeline, greatly reducing the likelihood for attack success.

2.4.4 Ransomware Mitigation

With ransomware's increasing prevalence in recent years, solutions specifically tailored to minimize ransomware's impact should be considered. Offline backups assist with system reconstitution in the event of a ransomware attack. Due to the generally static nature of ICS environment configurations, storing offline backups of OT systems should be performed. As an example, the configurations associated with HMI displays and the logic files of the PLCs go unchanged for long periods of time. Offline backups of these systems are crucial in the event your systems are tampered with, or reprogrammed. The infrequent back-up cycle of OT components results in minimal personnel time investment and the likely small amount of data results in inexpensive offline storage requirements. The system's IT components should be backed-up more often if possible. Even so, ransomware recovery methods for the IT components which automatically backup systems on an unwritable opal drive would quickly speedup recovery time from weeks to hours [14]. The capacity to greatly minimize the impact of theoretical attacker A's

ransomware or theoretical attacker B's PLC reprogramming would allow operators to quickly reassert positive control and situational awareness. With positive control re-established, the lye levels would be discovered and corrected.

2.4.5 Training and Awareness

Training which attempts to increase the cyber-hygiene of personnel should be implemented to minimize unwitting insider threats. Periodic tests which help to understand the current state of cyber-hygiene within an organization should be regularly practiced. A common example of this being phishing e-mail tests. Additionally, it is crucial to define and exercise a response plan which dictates the actions to be taken and by whom in the event of a cyber incident. Lastly, personnel should be aware of the threat that cyber has on these systems and how they should respond given certain indicators or scenarios.

2.5 Hardware Abstraction Research Approach

Exploring these realized and theoretical scenarios illuminates both the realized and unrealized threats of today and tomorrow. This exploration serves to guide researched approaches and provide success metrics to ensure effective solutions. While the low-cost mitigations detailed in section 2.4 strive to establish strong cyber boundaries and awareness which reduce the likelihood of an impactful compromise, heightened resilience is achieved through the ability to recover and continue operations despite a system compromise. However, an effective means of recovery does not eliminate the requirement for strong cyber boundaries. Both cyber boundaries and effective response and recovery methods are required and

complementary to truly achieve effective and holistic resilience. For example, without effective detection, it could prove difficult to decide if a response is required.

Hardware abstraction is the decoupling of hardware and software interdependencies which enables the flexibility necessary to respond and recover from realized threats at the device level. The resilience and cost savings benefits of hardware abstraction can be observed in the IT domain with a common example being the use of virtualization environments to detonate and study malware. In this example, the impact of a malware's effect is neutralized due to the lack of direct hardware access, the trivialization of user-commanded environment destruction and reconstitution, and the ability to easily revert to previous states. Hardware abstraction is commonly realized using virtualization and containerization. They each have benefits which best suit them for their respective use cases with virtualization being more secure and containerization being more scalable.

This flexibility applied at the industrial device-level within the ICS domain mitigates threats at the process level. The following table and subsections highlight how this flexible resilience mitigates the operational impact caused by both theoretical scenarios.

Table 2-4 Analyzing the Hardware Abstraction Approach's Applicability to the Operational Threats

Operational Impact	**Hardware Abstraction Approach Relevance**	
	Effort	Effectiveness
Control Manipulation	**1**	**3**
Data Destruction	**1**	**3**
Reduction of System Confidence	**1**	**3**
Delivery of Unsafe Water	**1**	**3**

2.5.1 Mitigation of Control Manipulation

Theoretical attack A's manipulation of control, paired with the limited situational awareness resulting from the execution of ransomware, is effective given its relatively minimal effort required. In this scenario, an approach utilizing hardware abstraction could ensure a trusted state and logic was deployed, resulting in expected process control. Moreover, variations of hardware abstracted processing instances with constrained or fail-safe operating limits could be deployed resulting in a control process able to continue the safe delivery of water while additional investigation and mitigation occur. This approach is further enhanced when combined with automation for timely process recovery despite the minimal personnel resources.

Theoretical attack B's PLC reprogramming as a method of covert control manipulation is highly effective and difficult to quickly respond to even if attacker motives are discovered. This difficulty results in longer malicious control manipulation which stems from manual processes associated with device reprogramming, an uncommon system action. Additionally, due to the attacker's network persistence, reinfection would be likely following reprogramming. Hardware abstraction's ability to quickly assume process control responsibility using a trusted programming logic and configuration enables prompt recovery. Additionally, hardware abstraction's flexibility supports the ingestion of rulesets preventing further attacker reinfection. These benefits are amplified by the security orchestration previously described in subsection 2.4.3.

2.5.2 Mitigation of Data Destruction

In theoretical attack scenario A, the deployed ransomware only propagated to the control system's IT infrastructure which is mitigated through the employment of the ransomware response technique previously outlined in subsection 2.4.4. However, it is a reasonable assumption that additional time along with the presence of device-level vulnerabilities could

result in device-level ransomware infection. Moreover, using a modification of the tactics found in theoretical attack B, the attacker could completely destroy the device's previous logical programming and configuration following the lye increase. This would eliminate the PLC's ability to correct the lye to nominal values. These device-level data destruction scenarios would be mitigated using PLC-level virtualization snapshots. Virtual machine snapshots are captured states which can be securely stored and exercised as a fall-back state for virtual instances. Using the virtualization approach, the system's programming and configuration would be decoupled from the process' hardware and I/O, enabling the capture and utilization of snapshots, and reducing the effort required to restore an affected process.

2.5.3 Restoring System Confidence

Not having confidence in the system's ability to perform its proper controls results in a lack of confidence in its process output. In both theoretical attack scenarios, any discovered anomalous behavior or attempted lye increase would impact the operators' and asset owner's confidence that their system will produce a non-hazardous output. This lack of confidence further contributes to impacted operations which ultimately affect the customers. A hardware abstraction approach which can be cryptographically assured and quickly defaulted to when there are suspicions of system compromise would be imperative to regaining confidence in the system and its outputs. This approach would facilitate the definition of a known and trusted configuration as an untouchable default which can be quickly employed via automation. Any assets suspected of compromise can be discarded or archived for later analysis.

2.5.4 Cost Savings and Uptime Benefits

Lastly, the hardware abstraction-based approach adheres to the ICS domain's justifiable prioritization of cost savings and system uptime. Abstraction from hardware results in a reduction in hardware costs and supports a reduction in the costs associated with vendor lock. Moreover, hardware abstraction enables additional low-cost logical redundancy and introduces the ability to hot swap process controls, reducing the downtime associated with program troubleshooting, system patching, and pushing configuration and logic modifications.

2.5.5 Ensuring the Delivery of Safe Water

The benefits of a hardware abstraction-based approach outlined in section 2.5 would ensure the delivery of safe water in both theoretical scenarios in the event enhanced cyber perimeters fail to prevent infection.

2.6 Conclusions

This chapter described the 02/05/2021 attack on the Oldsmar water treatment plant and how it serves as an example for the current state of many systems responsible for providing basic needs. Theoretical scenarios showed how minor changes could have resulted in a dangerous situation for thousands of people. This chapter's proposed mitigations increase the difficulty to execute these attacks while reducing their likelihood for success. Moreover, the adoption of virtualized and containerized PLC technologies enables the mitigation of effects which have an operational impact. The 2021-2022 Oldsmar annual budget contains only a single line directly relevant to this topic, "Continue network hardening measures around City's supervisory control and data acquisition at reverse osmosis water treatment plant and water reclamation facility." with no specifics describing how that will be accomplished or how success will be measured [16].

The shortcomings outlined in the research are not isolated to Oldsmar and should be viewed as a general reflection of the municipal control system landscape. A landscape which could become a lucrative target for asymmetric cyber warfare operations. This research hopes to be a motivator to augment existing systems with these mitigation concepts and serve as a potential influence for the requirements of the unbuilt systems of the future.

2.7 Future Work

The proposed low-cost solutions outlined in this research are effective at reducing the likelihood for system infection and subsequent operational impact. However, a reduction of compromise likelihood is not the elimination of it as the cyber threat landscape is expansive and constantly evolving. Therefore, the research approach outlined which utilizes hardware abstraction to enable the response and recovery of assets despite a cyber incident will be explored. The theoretical effectiveness of this solution should be proved in tangible ICS environments and its applicability to conform to the fundamental constraints of the ICS domain should be investigated. The subsequent dissertation chapter showcases research which defines this approach, formalizes an architecture, develops a technology, and subjects the approach to experimentation [1].

3 A Containerization-Based Backfit Approach for Industrial Control System Resiliency

3.1 Introduction

PLCs are prominently used in industrial control systems for the operation of critical infrastructure. The prevalence of PLCs can be observed in the water treatment, manufacturing, energy, and natural gas distribution industries. PLCs are ruggedized, embedded, real-time systems which execute programmed logic that interfaces with physical sensors and actuators by reading and generating digital and analog signals. These signals are sent and received via the PLC's input/output (I/O) interfaces connected to the control system. It is this cyber-physical relationship which drives critical processes and comprises the ICS.

Due to the highly critical nature of the ICS domain, installed PLCs are expected to continuously operate for extremely long periods of time. These devices and their supported protocols often lack authentication and encryption as real-time operation with minimal interruption is prioritized. With long device replacement cycles, little downtime allotted for updates, and a lack of authentication these devices are prime targets for cyber adversaries. When targeted, the process for recovering ICS assets is highly manual and lengthy.

Depending on the size of the ICS and the cyberattack's scale, execution of these manual procedures can take days to weeks. Additionally, due to a reliance on identical hardware replacements, a recovered system's attack surface will be identical. To an attacker with a strong foothold on an ICS network, even these lengthy procedures are likely to fail to mitigate the situation. An example of this can be seen in the Stuxnet ICS rootkit which used very low-level device access to thwart any attempt to reflash the logic or firmware. Stuxnet also had a presence on the engineering maintenance laptop, meaning the configuration of a new device could result

in reinfection [28]. PLCs represent a cyberattack surface that can be made more resilient through rapid response, and recovery capabilities.

3.1.1 Contributions

Containerization bundles software and dependencies into lightweight and portable container images. Containers operate using minimal computational overhead and provide security through application isolation. In addition to process isolation, containers can be quickly created and destroyed. This flexibility stands to introduce high levels of resiliency to the ICS domain through the swift reorganization of live logical assets to ensure continuity of control operations. This research proposes that containerizing a PLC's runtime while maintaining its physical process control applies many of the resiliency benefits afforded by containerization to ICS environments. Moreover, this research proposes an architecture which leverages containerization technology to enhance preexisting control systems, resulting in a backfit solution.

The main contributions of the research are:

- To the authors' best knowledge, this is the first work to research and develop a hardware interpreter capable of maintaining strict container isolation while allowing physical outputs to be generated by a containerized PLC's runtime state.
- To the authors' best knowledge, this is the first work to research and implement an I/O multiplexing container host, facilitating the approaches' applicability to preexisting control systems.
- This research also applies these novel technologies in an orchestration architecture resulting in automated ICS process recovery.

- The proposed approach was evaluated via experimentation on physical ICS testbed environments.

3.2 Related Work

Containerization has been previously applied to ICS networks. Previous research applied containerization for robustness, security, and ease of maintenance [29]. However, this previous approach did not consider augmenting existing systems with support, nor did it leverage automated response actions to mitigate active threats which are addressed in this research. ICS containerization for attacker deception has also been researched [30]. Divergent to that research, this research aims to leverage containerization, not as a decoy, but as a means of active system and process recovery. Research in time-sensitive computing also indicates that containers can be configured to meet the performance demands of many ICS processes [31].

An additional related area of ICS research is the hardware-agnostic software PLC with a prominent example being OpenPLC [32]. Software PLCs currently support research thrusts of low-cost ICS training and development, high-fidelity ICS network modeling and simulation, and feasibility testing of ICS security concepts. This research will leverage OpenPLC to provide the PLC runtime environment.

3.3 Respond and Recover: A Containerization Approach

Unlike virtualization, traditional containerization does not leverage a hypervisor, instead opting for kernel isolation for container and host separation. While not providing the same level of secure isolation as a hypervisor, containerization provides increased performance and lower computational overhead compared to virtualization while providing more isolation than running

an application on bare-metal hardware. By containerizing a PLC's run-time while maintaining physical process control, ICS networks can adopt the resiliency benefits provided by process isolation, dynamic orchestration, and high-speed process recovery. Additionally, the financial benefits from the reduced downtime and hardware costs further incentivize approach adoption. Once equipped with these resiliency technologies, ICS networks can automatically mitigate cyber-threats and quickly recover critical processes.

3.3.1 PLC Containerization

This approach relies on four fundamental technologies, first is the containerized PLC. This research will containerize the open-source PLC implementation OpenPLC to provide the PLC runtime environment. The containerized PLC must have the ability to generate physical control signals to interface with its environment. As a result, the containerized PLCs are hosted on or have access to a device with direct connections to the ICS endpoint's sensors and actuators. Due to the hardware segmentation of unprivileged containerization, a hardware interpreter was developed. This hardware interpreter was programmed in Python and ran on the host system. The hardware interpreter would query the state of the containerized PLC and use an I/O map to output the appropriate state to the control system while maintaining the isolation benefits of containerization. This query used the PyModbus Python package to generate Modbus/TCP read commands sent over the local docker interface [53]. Therefore, all containers ran unprivileged, resulting in increased container isolation. A Raspberry Pi-based host was used for experimentation. Figure 3-1 showcases the implementation architecture and data flow.

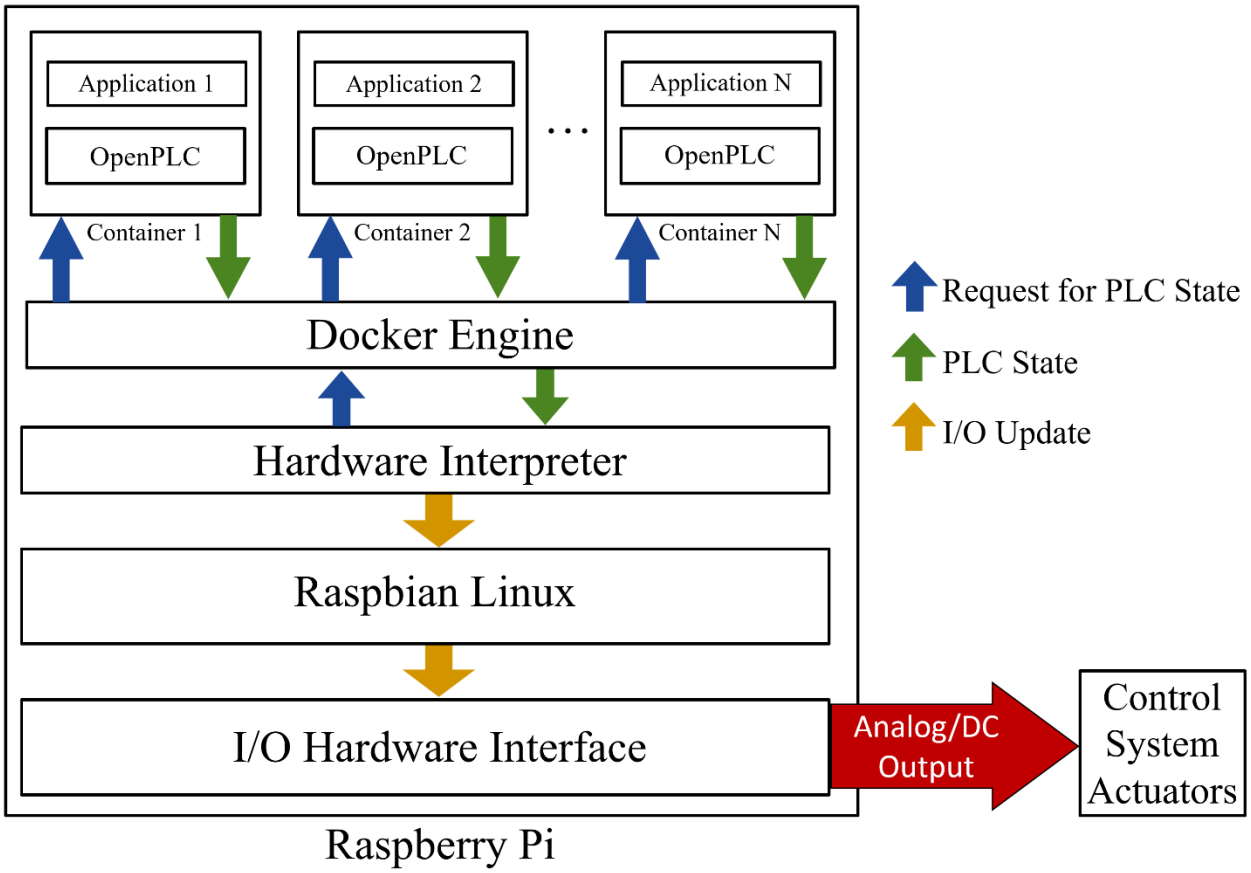


Figure 3-1 The Containerized PLC Architecture Supporting the Generation of Physical Output

Each container PLC's state queried by the hardware interpreter was transferred using the Modbus/TCP protocol over a network internal to the Raspberry Pi. The containerized PLC's support of this protocol allows them to interface with traditional PLC networks which use this protocol. This networked state exchange could be distributed such that the containerized PLCs can run on a remote host with the PLC state exchanged to the hardware interpreter physically co-located with the control system.

3.3.2 I/O Multiplexing

Second, is the ability to quickly change the physical and logical processes responsibility from traditional PLC to containerized PLC. For physical processes, this can be done using the

containerized PLC's host as a passthrough system that can take control when prompted. To accomplish this, the output of the physical PLC is wired to the input of the container host. The container host's output is wired to the control system. The hardware interpreter monitors the output signals originating from the physical PLC using the UniPi's SysFS driver which exposes I/O control to files in the Raspbian filesystem. The hardware uses this driver to monitor the input pins which represent the output signals of the physical PLC and either passes the values to its UniPi chassis' output pins or ignores the values, instead using the state of its hosted container PLCs to generate physical output. For the logical processes, networking rules and flows can quarantine an infected PLC and designate all traffic to be rerouted to its containerized replacement. Figure 3-2 shows an expanded architecture which supports rapid I/O switchover.

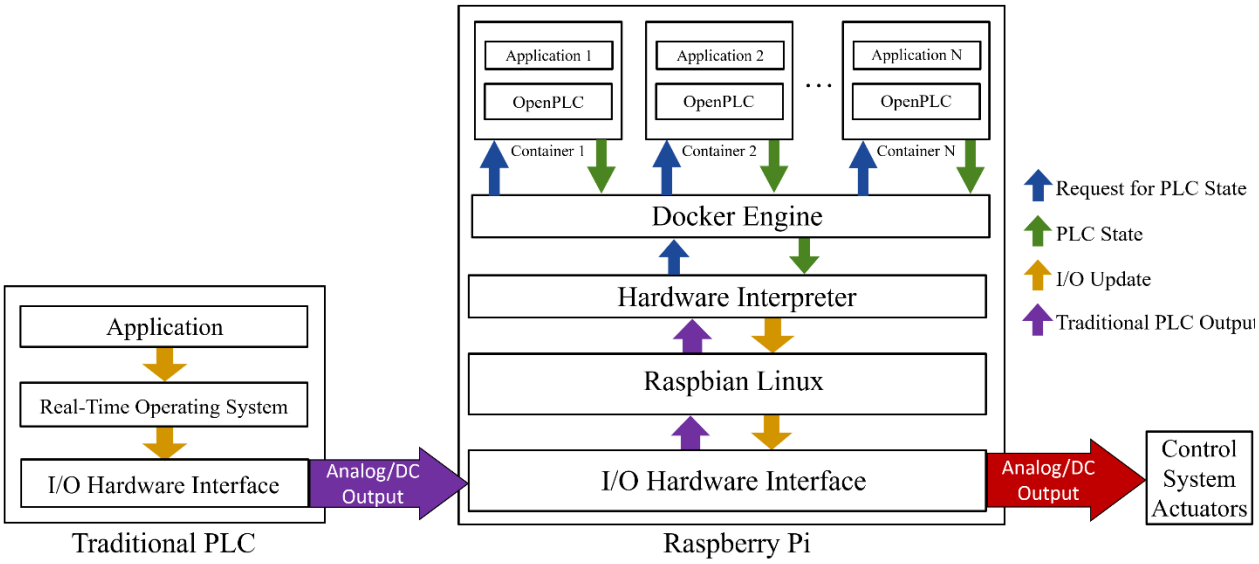


Figure 3-2 The Containerized PLC Architecture with the Traditional PLC Passthrough Capability for Supporting Preexisting Systems

One issue with this approach is the container host is now a constant point of failure regardless of the PLC type responsible for the control process. This can be easily remedied by implementing

the preexisting PLC's output signals in a normally closed (NC) circuit which only opens when output from the container host's I/O interface energizes. This results in the preexisting PLC having direct electrical control of the system unless the containerized PLC host generates control system output which would then take precedence. Despite the possible reliability benefits of the NC architecture, the hardware interpreter passthrough architecture was chosen due to the simpler installation, smaller hardware footprint, and the added situational awareness gained from ingesting the physical PLC's outputs into the container host. The following flow diagram in Figure 3-3 showcases the logic of hardware interpreter's processing and I/O multiplexing. A code sample is provided in Appendix B.

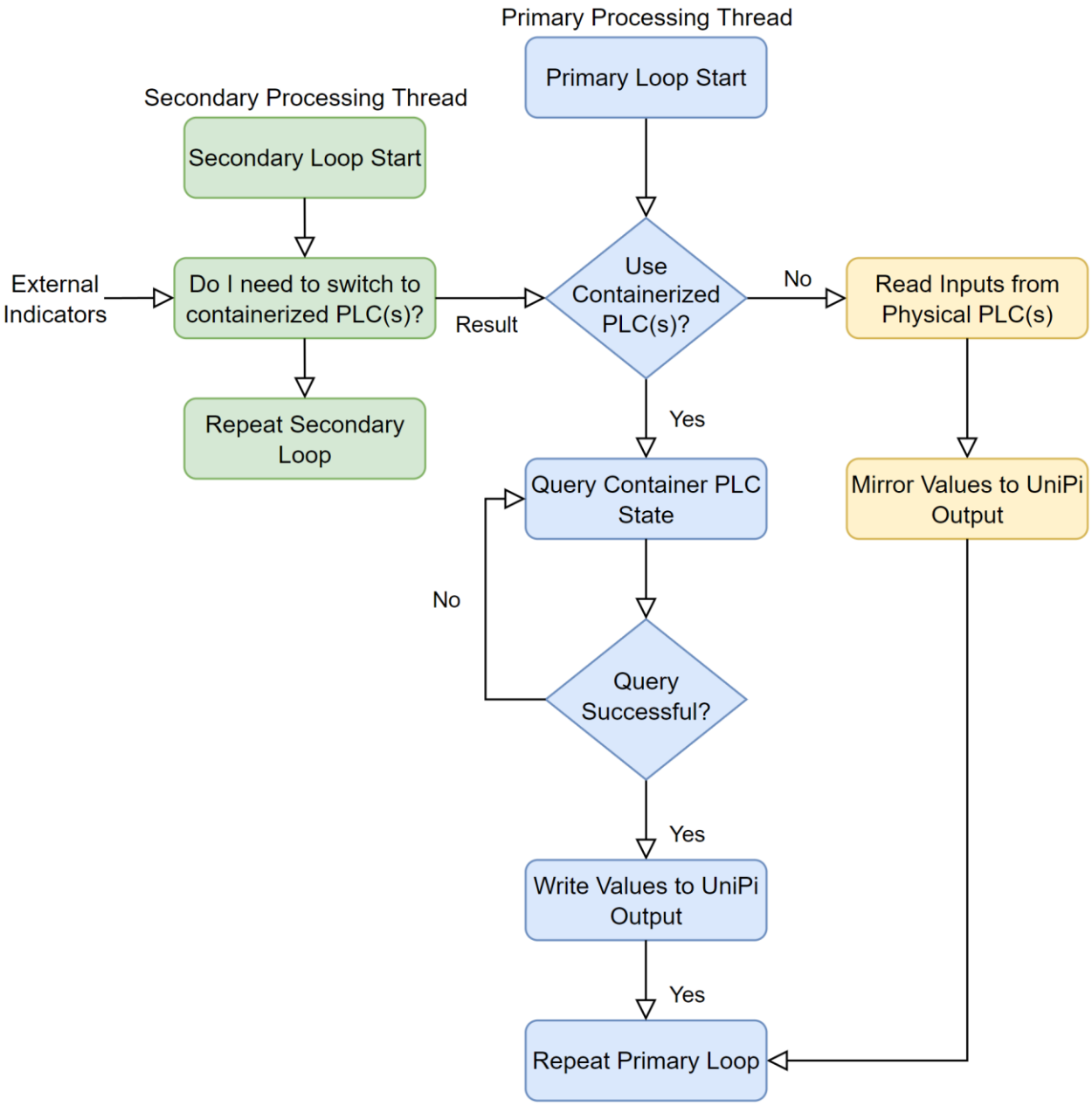


Figure 3-3 The Process Execution Flow of the Hardware Interpreter

3.3.3 Orchestration

The third technology is the automated management and orchestration of the containerized PLCs. Recent advancements facilitating the automated and semi-automated responses of cyber-incidents enhances the speed and efficiency of this approach while moving away from the

traditional castle model approach to ICS cybersecurity [33]. This castle model consists of strong cyber barriers such as firewalls and multi-factor authentication akin to castle walls and relies on an assumption that if a system's walls are secure enough, the likelihood of a compromise is heavily minimized. While a strong cyber perimeter is the first line of defense against malicious adversaries, an approach that can mitigate attackers inside this perimeter would ensure that critical assets remain steadfast.

An orchestrator was developed to ingest indicators of compromise and would send commands to the hardware interpreter prompting a change from physical PLC I/O passthrough mode to containerized PLC output. This aimed to show that an automated response could be taken to have an actionable effect on the ICS leveraging containerization technology.

3.3.4 Anomaly Detection

Lastly, the fourth technology is the anomaly detection system which monitors the network and its devices, influencing the decision to transition to containerized PLCs. This topic is very well researched and varying approaches are highlighted in this chapter's experimentation section. As such, it will not be a focal point in this chapter. A simple anomaly detection script was developed to trigger the system's failover from physical PLC to containerized PLC. The anomaly detection script would leverage internet control message protocol (ICMP) pings to heartbeat the primary physical PLC once a second. This one-second interval was chosen to not disrupt the network. When a response is not received, the anomaly detection script sends a message to the hardware interpreter to cease the I/O passthrough mode and instead use the state of the containerized PLC to energize the container host's I/O resulting in physical system actuations.

3.4 Experimentation

This section details the technology installation and experimentation which occurred in the water treatment and water chilling ICS testbed environments.

3.4.1 The Water Treatment Experiments

The first experiment took place on a water treatment testbed. Water treatment represents a critical infrastructure sector which poses a risk to human health if compromised. The testbed leverages four pumps which operate the flow of water through the system. Process control is provided by three Allen Bradley PLCs and a windows-based HMI. The Allen Bradley PLCs and HMI transfer data and commands using Modbus/TCP, a prominent communications protocol found in ICS networks. The Allen Bradley PLCs used a ProSoft MVI56E-MNET network interface card to support Modbus/TCP [35]. One Allen Bradley PLC is configured to be the primary logic controller, its analog outputs control the main feed pump and discharge pump. The primary PLC also passes commands to the two secondary controllers which are each responsible for a subprocess of the system. One secondary PLC is responsible for the backflush subprocess pump, backflush is an operation mode which runs the plant in reverse and is commonly used for system maintenance and cleaning. The other secondary PLC is responsible for the chemical pump, the pump which determines the rate chemicals are added to the water treatment mixture. All pumps leveraged a 4-20mA analog control signal produced by the PLC to determine their speed with 20 mA equating to full pump speed. A picture of this system can be seen in Figure 3-4. This experiment aimed to prove the feasibility of leveraging containerized PLCs to control a cyber-physical system (CPS).

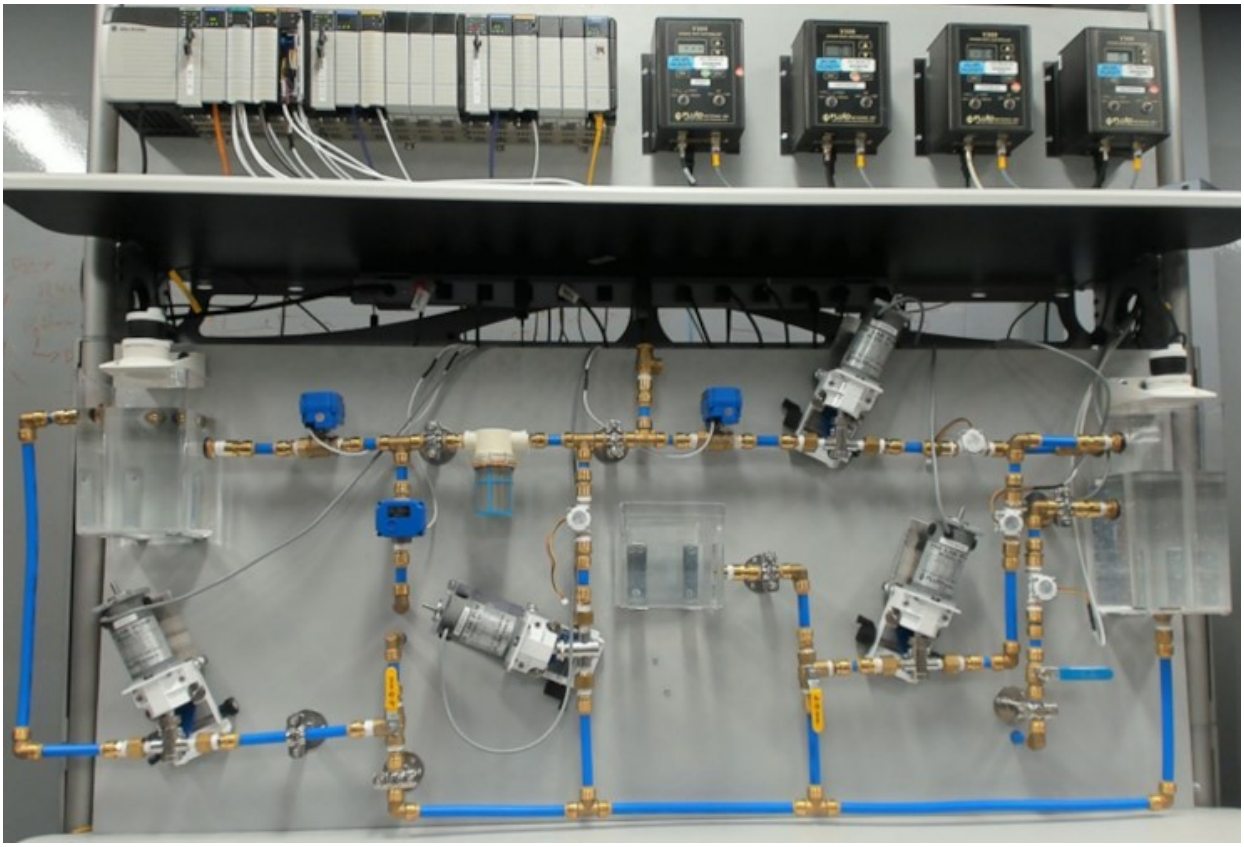


Figure 3-4 The Water Treatment Testbed

The containerized PLC was hosted on a Raspberry Pi model 3B+ and leveraged a WidgetLords Electronics Pi-SPi-2A0 Raspberry Pi analog output interface [34]. This interface allowed the Raspberry Pi to programmatically generate 4-20 mA signals to control the testbed's pumps.

The experiment began by running the standard testbed and capturing network traffic to later compare with the testbed controlled via containerized PLCs. After baselining network behavior, the containerized PLC was installed in place of the physical primary PLC. This container ran a logically equivalent OpenPLC run-time with an identical network configuration. The hardware interpreter ran on the host Raspberry Pi and generated physical mA outputs based on the container PLC's logical state. The water treatment system was operated again under containerized PLC control. While running on the containerized PLC, the system's behavior was

noted. The system did not exhibit any physical anomalies, and the operator HMI control functioned identically. Additionally, the two secondary physical PLCs interfaced with the containerized PLC normally and produced correct output. Network captures were taken under this configuration for comparison.

Despite producing the expected output for each given input, the variables within the logical application were initialized at zero. As a result, operator input was required to start the system. In a step towards mitigating this behavior, an additional container image of the primary PLC was created to have prepopulated values determined to be a safe fallback system state. For this experiment, the safe fallback state was defined as a normally flowing system with a pump speed of 15%. The prior test was repeated, this time using the containerized PLC with the prepopulated state. As expected, when the containerized PLC was given system control, its prepopulated state ran the physical water treatment system at 15% speed. While this approach did not initialize the system at zero, the state of the physical system did not infer the state of the containerized PLC prior to its instantiation. The system was then altered to also remove the physical backflush PLC and attach an additional analog output to the containerized PLC host giving it the ability to control the backflush PLC's pump. The interface was added by daisy-chaining an additional WidgetLords analog output board to the system's current board which can be seen in Figure 3-5.

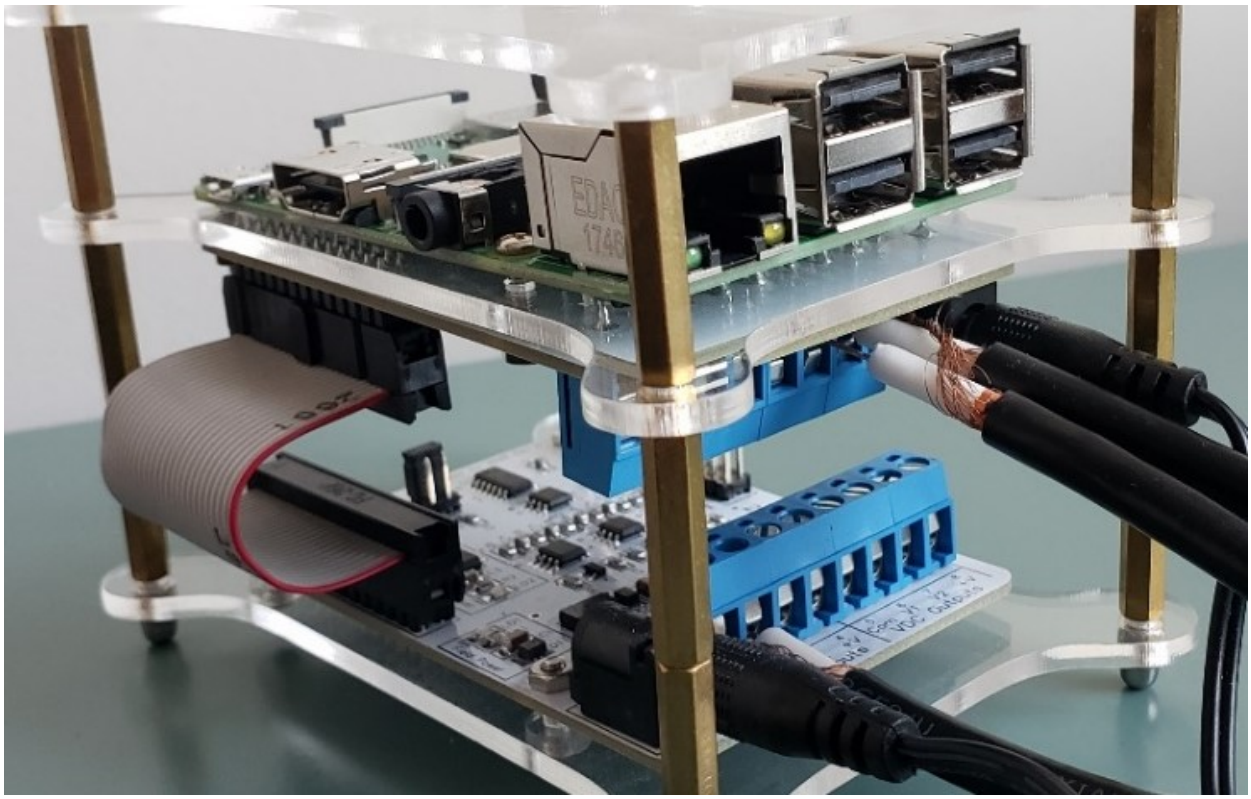


Figure 3-5 The Raspberry Pi Container Host with Two 4-20 mA Output Boards

This aimed to show that two PLC containers could run on a single host and maintain proper process control. Under this new configuration, the system maintained normal functionality and the containerized PLCs running on the single Raspberry Pi host were able to properly execute both the primary and backflush processes. Despite observing expected control system functionality, the daisy chained I/O solution of the container host was not scalable and would struggle to support control systems with large diverse I/O requirements. A limitation addressed in the subsequent experiment.

3.4.2 The Water Chiller Experiments

Additional experiments aimed to demonstrate a completely automated recovery while maintaining system state, address the issue of I/O scalability, and introduce a system threat. This

second experiment leveraged a water chiller testbed. Water chilling is a crucial process in various infrastructure systems, providing a constant supply of cooled water to maintain the temperature of a process. Due to chilled water supporting many other critical infrastructure processes, it makes for a good side-channel target due to the cascading effects a lack of cooling can have on a given system. This testbed used two Allen Bradley PLCs to read system inputs and produce appropriate outputs. In the case of this system, outputs included running a forward pump and drain pump to cycle the water based on system temperature and water levels. Heating coils were perpetually heating the system to simulate a load generating heat to be dissipated. This system can be seen in Figure 3-6. Unlike the outputs of the previous water treatment testbed which were granular and could support multiple speeds, the chiller testbed's outputs were binary meaning the pumps and valves were either fully on or fully off. This is due to the increased precision required for the water treatment testbed's chemical mixing.

Figure 3-6 The Water Chilling System with Sensors, Actuators (left), and Controllers (right)

Although two physical PLC chassis were present on the system, they were running in a remote I/O configuration. With this configuration, one PLC logically drove the system while the other served as an additional bank of I/O interfaces. Despite not having granular input, the chilled water testbed had more I/O making the initial experiment's approach to I/O via daisy-chaining impractical. Due to the increased number of I/O, we shifted the container host from the Raspberry Pi with the WidgetLords interface boards to a UniPi L513. The UniPi was selected due to its additional I/O interfaces and flexible Raspberry Pi computing platform [36]. Figure 3-6 shows the system's two PLC chassis and the blue UniPi.

During the first water treatment testbed experiment, the container host had to be manually installed to support a switch to containerized controls. To mitigate this by automating the

physical response of the containerized PLC, the container host was placed in-line of the physical PLC's I/O. This allows the containerized PLC to assert physical system control from a compromised PLC by choosing to ignore the physical PLC's outputs, instead opting to leverage its own output for physical process control. This implementation can be seen in Figure 3-7. All system inputs were forked to go to both physical and containerized PLCs. The containerized PLC was programmed to be logically equivalent to both PLCs chassis and the hardware interpreter was updated to support this new containerized PLC and new UniPi hardware platform.

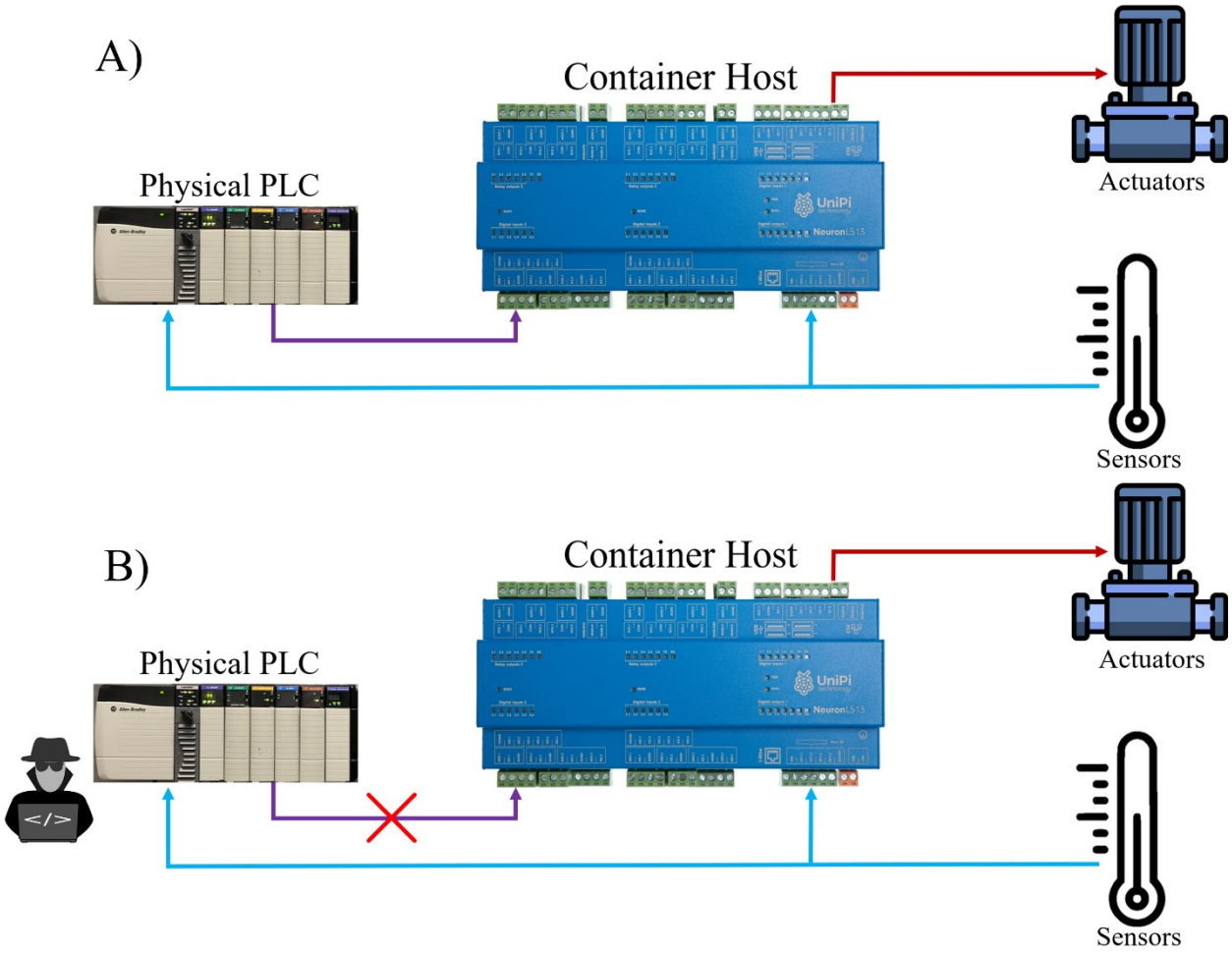


Figure 3-7 A) A UniPi PLC Container Host and Its I/O Installation, B) A UniPi PLC Container Ignoring the Output of an Infected PLC

The containerized PLC was configured to execute its logic regardless of its control over the physical I/O. This allowed the containerized PLC to maintain process state by reading system inputs while operating in passthrough mode. Once passthrough mode was disabled by the orchestrator, the containerized PLC's state which was previously influenced by system inputs would be in control of the physical system. One shortcoming of this approach is a lack of state validation as a previously corrupted physical state could be inherited by the containerized PLC. This can be mitigated by using a containerized PLC with predefined fail-safe bounds of logical values. Additionally, other research has applied distributed ledgers to provide cryptographically assured ICS states [38]. There are no technical limitations for applying these techniques in conjunction with the containerized PLCs.

A scenario was executed which simulated a denial-of-service threat to the physical PLCs. This scenario involved cutting power to the physical PLCs while they controlled the system. When this occurred, the anomaly detection script did not receive its expected heartbeat and consequently instructed the hardware interpreter to go from pass-through mode to containerized PLC mode. The control system froze for a one-second duration due to the anomaly detection script's one-second heartbeat interval. Upon recovering to the containerized PLC's runtime, the system performed as expected by properly processing all system inputs and producing correct outputs.

The observed one-second delay would be unacceptable for most industrial applications. However, this one-second delay was an artifact of the simple heartbeat-based anomaly detection and an implementation leveraging passive deep packet inspection (DPI) [39] or periodic active scanning [40] would not have such a delay and could detect more complex attacks. To quantify the maximum speed the control system could go from pass-through mode to containerized PLC

mode, an additional experiment was performed where the networked heartbeat-based anomaly detection was substituted for monitoring the physical PLC's output. When the output corresponding to the testbed's running state lost power, the anomaly detection script would switch to containerized PLC mode. Data measurements were taken to quantify this time delta, but the mode switch occurred faster than our monitoring hardware's 1µs polling rate.

PLCs are purpose-built to operate real-time controls, therefore an additional experiment to benchmark a containerized PLC running on the UniPi in comparison to a physical PLC with maximum performance settings was executed. This was done by measuring the latency between pressing the run button and illumination of the green light which indicated the chiller was in run mode. This latency was measured using the LabJack T7, which is a device that can read and generate control signals [37]. The red LabJack T7 can be seen in Figure 3-6 installed next to the UniPi. The LabJack was wired into the system such that it could produce a voltage to simulate a button press and subsequently determine if the wire controlling the green light had a voltage, resulting in illumination. Both the reading and creation of control signals can be programmed using the device's support of the Lua scripting language. A Lua script was written to generate a signal to match a button press, this is when a timestamp was taken. Upon reading a voltage indicating light illumination, an additional timestamp was taken. The difference between the timestamps is calculated and saved in an array. The script loops 500 times and saves the data upon completion.

3.5 Experimentation Results

The four observational questions below were defined to establish and verify system behavior:

1. Are reported sensor values reflecting expected values?

2. Is the physical process behaving as expected?
3. Are system network communications functioning as expected?
4. Does the operator interface exhibit nominal behavior?

These questions were supported by actions which would observe and stimulate the system, providing insight into system behavior. Both testbed environments were subjected to these questions prior to modifying the systems with the containerized PLCs. This was done to ensure the system was functional prior to experimentation and these initial question answers and system observations served to establish a baseline of nominal system behavior to later compare against. The system's observations were noted during experimentation and are shown in Table 3-1 below.

Table 3-1 Observations of System Characteristics Throughout Experimentation

Experiment	**System Characteristic Observations (Yes/No)**			
	Are reported sensor values reflecting expected values?	Is the physical process behaving as expected?	Are system network communications functioning as expected?	Does the operator interface exhibit nominal behavior?
The First Water Treatment Experiment	Yes	Yes	Yes	Yes
The Second Water Treatment Experiment	Yes	Yes	Yes	Yes
Water Chiller Experiment	Yes	Yes	Yes	Yes

The data resulting from the water treatment system's network traffic was analyzed to compare the latency and network behavior of the physical PLC and containerized PLC. Using the

Wireshark filter “modbus.responsetime” returns the time it takes for a networked device to receive, process, and subsequently respond to a Modbus/TCP request. The graphs shown in Figures 3-8 and 3-9 chart the maximum response time observed in 100ms intervals plotted throughout the duration of the captures. Figure 3-8 shows the maximum response times of the Allen Bradley PLC while Figure 3-9 shows the maximum response times of the containerized PLC. The packets monitored for response time values were Modbus/TCP commands exchanged with the secondary PLC. Communications with the secondary PLC were used as a benchmark as opposed to the HMI due to the HMI’s windows operating system which is less deterministic than the secondary PLC which leverages pre-programmed cycle times and a real-time operating system.

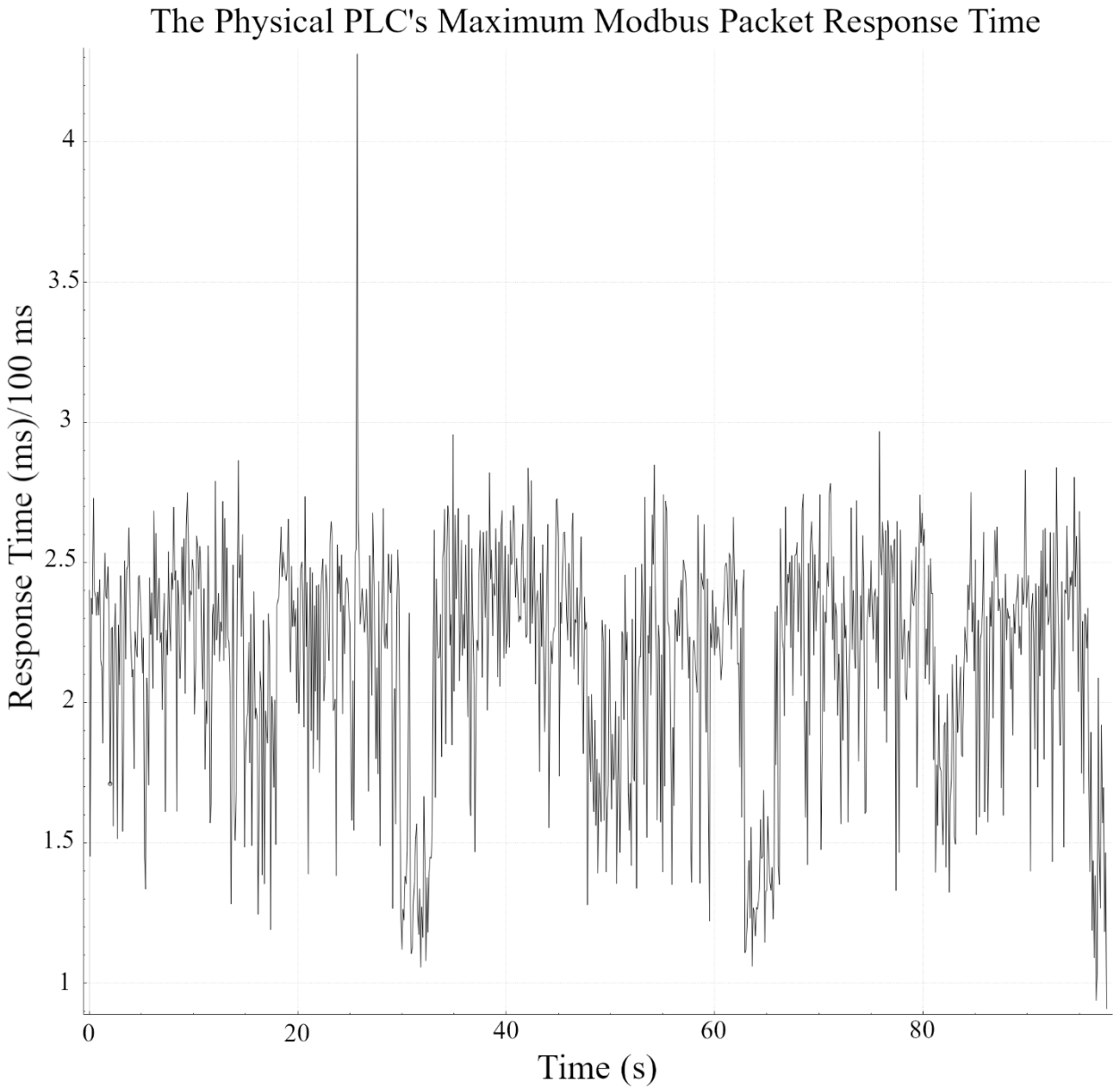


Figure 3-8 Maximum Modbus/TCP Response Latency in Milliseconds for the Physical PLC

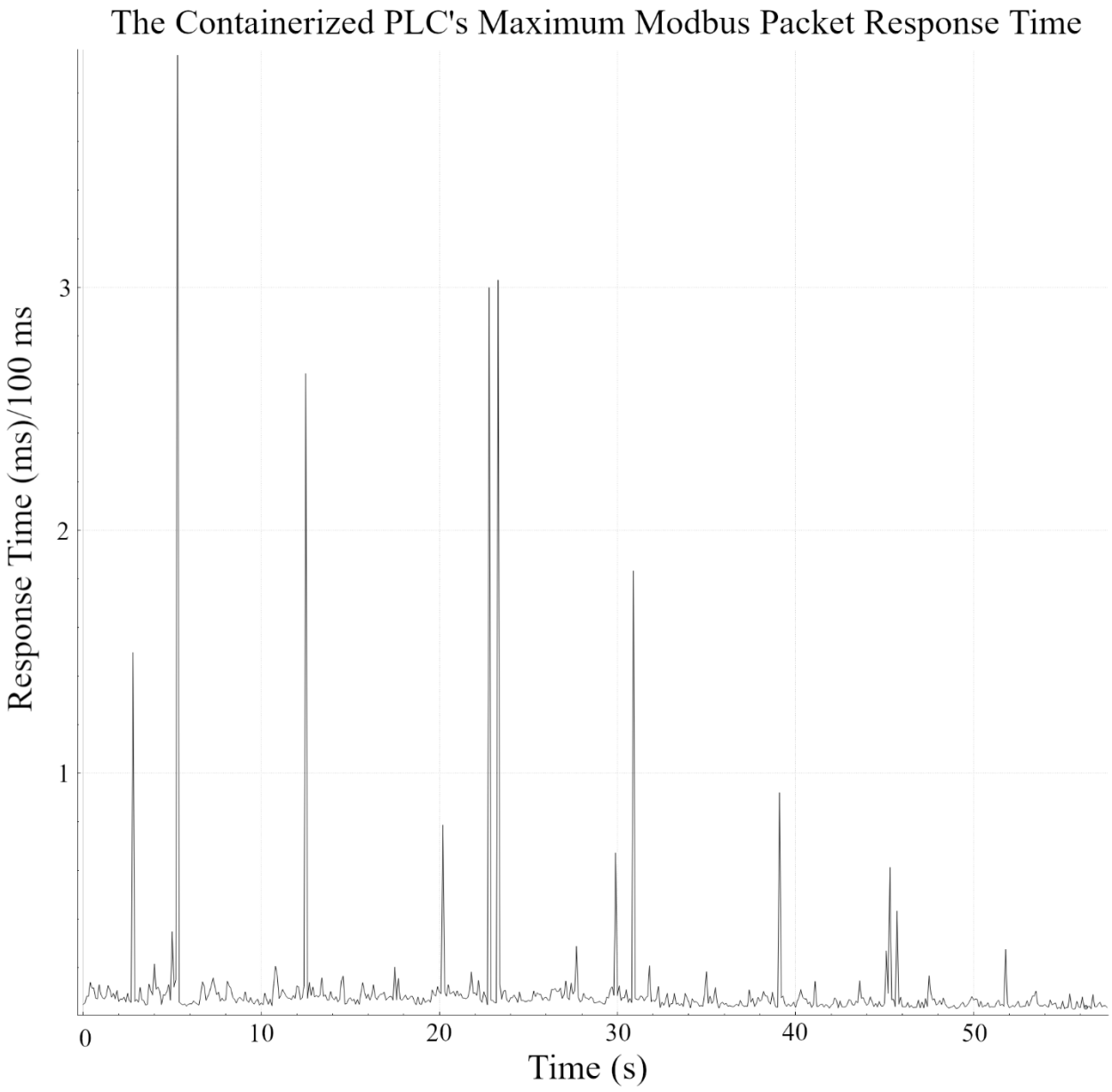


Figure 3-9 Maximum Modbus/TCP Response Latency in Milliseconds for the Containerized PLC

The maximum value was plotted due to the high potential impact a large delay could have on an ICS. These graphs show the containerized PLC's lower response time compared to the physical PLC, showcasing a shorter time between request and response. Table 3-2 compares additional network behavior statistics between the physical and containerized PLCs. The data shows the highest delay in response time occurred by the physical PLC. Additionally, the average

Modbus/TCP response time value was much lower on the containerized PLC. Lastly, the packets per second (pps) calculated for both PLCs show the containerized PLC communicates more often than the physical PLC. This could be due to the faster response time allowing more follow-up requests, or slight variances between the two PLCs' configurations.

Table 3-2 The Network Behavior of Both PLC Types

Communications Attribute	**PLC Type**	
	Physical	***Container***
Average Modbus/TCP Response Time (ms)	1.30795ms	0.05023ms
Maximum Observed Response Time (ms)	4.313ms	3.956ms
Packets Per Second (pps)	98.813 pps	114.228 pps
Minimum Packet Length	78 bytes	78 bytes
Maximum Packet Length	109 bytes	109 bytes
Average Packet Length	85.22 bytes	86.39 bytes

Table 3-2 also shows the packet length of both PLC approaches. Analyzing the packet lengths show the containerized PLC did not send any packets that were abnormally sized. Additionally, the packet contents were analyzed, and no discrepancies were observed between both devices.

The data taken from the chilled water system was analyzed to compare the latency of a full control loop. The control loop in this context being a simulated user button press, logically processing this input, and outputting the appropriate hardware actuation. Figures 3-10 and 3-11 plot the latency of 500 samples of both PLC approaches. These graphs show the containerized PLC's higher control loop latency when compared to the Allen Bradley PLC. Initial results

displayed in Figure 3-10 show an average of 24.616ms for the containerized PLC and 3.003ms for the physical PLC. Additionally, the data shows inconsistent containerized PLC latencies when compared to the physical PLC. Figure 3-11 showcases a secondary test where the containerized PLC runtime and hardware interpreter were given real-time kernel priority. In this test, the containerized PLC performed with an average latency of 18.118ms while the physical PLC averaged 2.856ms. Not only did the containerized PLC latency decrease with this configuration modification, but the latency was much less sporadic. This more consistent latency results in more reliable process controls. The 18.118ms latency was likely caused by a few reasons, primarily the OpenPLC instance was configured with a 20ms cycle-time. A cycle-time is a PLC's scheduled time to ingest I/O updates, complete execution of program logic, and produce resulting output signals. For reference, the Allen Bradley 1756 series PLC has a default cycle time of 10ms configurable up to 2,000,000ms [41]. Additionally, the Siemens S7-1200 has a default cycle time of 150ms [42]. Given the containerized PLC's performance, it can support industrial common processes requiring a cycle time of 20ms or greater. Additional research to further reduce the latency of the containerized PLC is outlined in the future work section of this chapter.

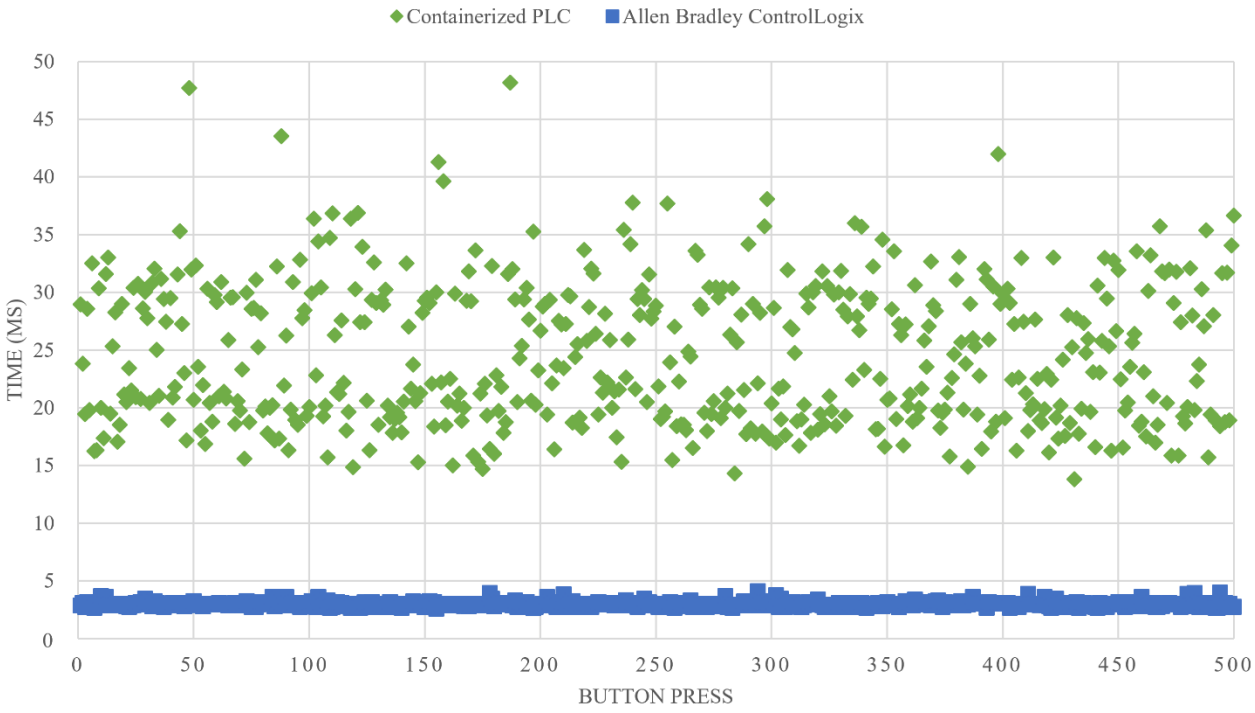


Figure 3-10 Latency Data from Input to Output for the Traditional PLC and Unoptimized Containerized PLC

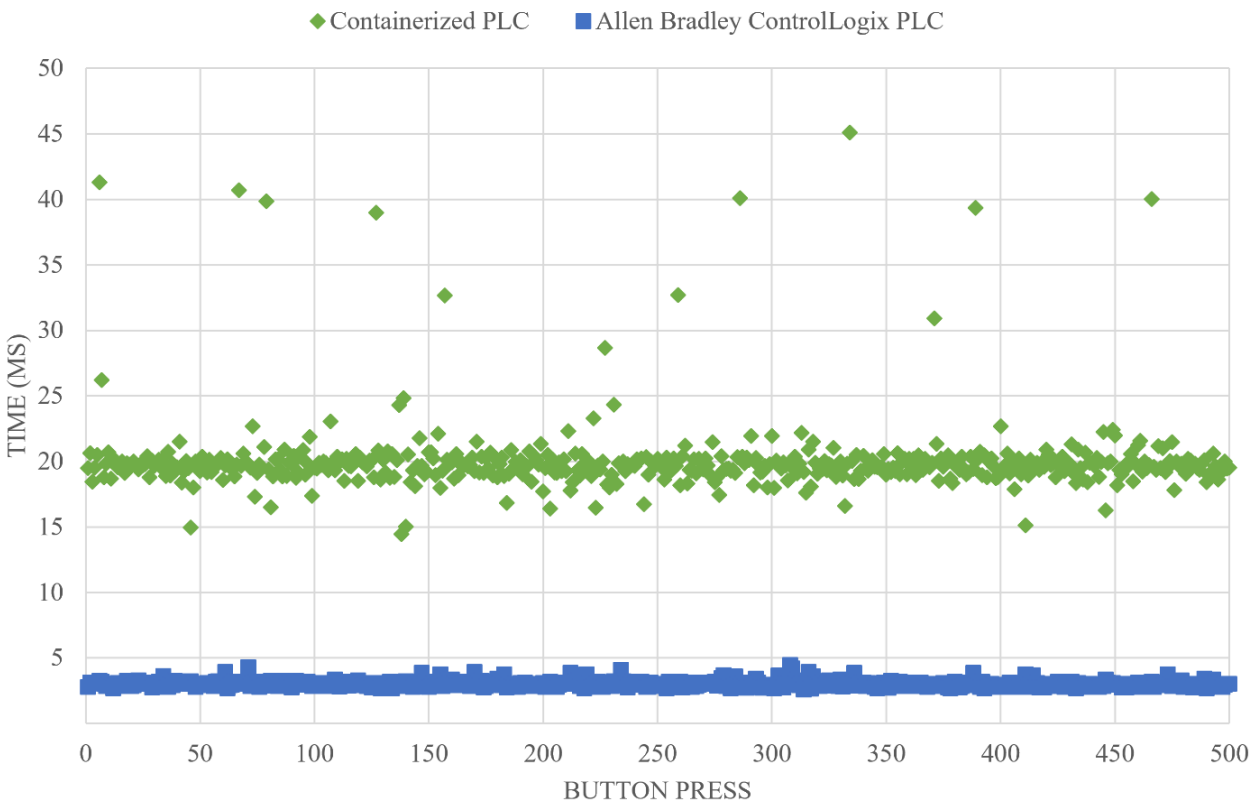


Figure 3-11 Latency Data from Input to Output for the Traditional PLC and Optimized Containerized PLC

3.6 Conclusions

This research proposed an approach for applying increased ICS resiliency through the containerization and orchestration of PLCs. This approach augmented traditional ICS environments and proved via experimentation that control system applications with soft real-time requirements could be supported. Experimentation showed the approach could quickly mitigate a system threat and ensure critical assets remain operational. No anomalies in the physical processes were observed and the differences in network behavior were minimal and inconsequential.

3.7 Future Work

A shortcoming of this approach which requires additional research to potentially mitigate is the manual generation of containerized PLC logic. This process must be automated for scalable widespread system adoption. This manual logic conversion also introduces the potential for human error. A system to automatically ingest logic samples and produce containerized PLC variants should be researched. Additionally, an approach should be researched which can automatically validate the logical equivalence of both physical and containerized PLCs.

Additional testing to optimize the performance of the containerized PLC could be performed. Leveraging a preemptive real-time kernel would likely reduce the button press latency by a considerable amount [31]. Additionally, recoding the hardware interpreter from Python to C would likely improve latency and consistency. Additional performance benefits could also be obtained by hosting the containerized PLCs on hardware with more computational power. Experimentation which quantifies the maximum number of containerized PLC instances a host can support while maintaining correct process control should also be performed.

The cryptographic attestation of PLC images and configurations could be researched to further ensure the integrity of the containerized PLCs. An image's failure to attest could infer the actions of the orchestrator. To that end, increasingly complex system threats and subsequent orchestration actions could be investigated to explore limits of orchestration as applied to the ICS domain. Lastly, this research approach was installed, tested, and validated on a United States Coast Guard maritime platform [6].

The research presented in the following chapter addresses many of these shortcomings. Firstly, by evolving the approach to utilize existing logic and configuration files, we eliminate the requirement to manually translate any previous logic which reduces the likelihood for user error.

Moreover, this evolved approach can utilize existing hardware and wiring to further reduce installation costs, downtime, and user error. Additionally, the evolved approach utilizes virtualization as opposed to containerization to increase the cybersecurity of the approach through greater isolation between guest and host. Nevertheless, the relevancy of the containerization utilized in this approach would be more applicable in environments with limited computing resources or when hosting a large number of instances numbering in the thousands due to containerization's low computational overhead and faster deployments. This evolved approach also utilizes a combination of orchestration and software defined networking (SDN) to enhance the response and recovery action space to include network reconfiguration in conjunction with PLC reconstitution. Lastly, this evolved approach utilizes a more capable virtualized PLC host which offers greater performance and additional cryptographic features not present on the host utilized in this experiment. These cryptographic features support the cryptographic attestation of the virtualized PLC and guarantee its trustworthiness.

4 A Backfit Approach for Trusted Virtualization-Based Programmable Logic Controller Resilience

4.1 Introduction

PLCs are real-time systems which receive inputs from sensors, execute pre-programmed logical routines, and energize outputs that ultimately drive physical actuators. These devices and their control loops operate a diverse range of physical processes, supporting various ICS and critical infrastructure sectors to include the energy, chemical, manufacturing, water, and wastewater sectors. Given the observed cyber targeting of these significant systems and processes, there is a compelling need to research methods that increase their resilience against cyber adversaries. Additionally, cost and uptime requirements often result in sparse upgrade cycles within the OT domain. Therefore, research considerations that explore approach applicability for existing proprietary systems should be made. This chapter proposes and demonstrates an approach which utilizes virtualization and trusted computing to enhance OT systems. These enhancements enable these systems and their processes to be more resilient, flexible, secure, and cost-effective.

4.1.1 Contributions

Virtualization generates a guest environment segmented from the host machine's hardware using a hypervisor to interpret and allocate its available computing resources. This segmentation provides several benefits. Firstly, a lack of reliance on a specific host results in a more dynamic computing environment as a virtual environment can rapidly change hosts as needed to ensure maximum uptime. Secondly, the isolation between guest and host can mitigate malicious processes from spreading to host hardware. Lastly, the hardware abstraction that virtualization

provides is cost effective compared to installing and maintaining dedicated hardware for each process which could be virtualized. For example, the ability to virtually test and seamlessly merge software updates and configuration changes with little to no downtime could prove impactful to these systems. This chapter proposes the virtualization of PLCs to enable these benefits within the OT domain while supporting existing systems and the domain's unique operational constraints with the envisioned future being a more dynamic, trusted, and cost-effective cyber-physical domain. Following an extensive literature review, the authors assert the main contributions of this research are:

- To the authors' best knowledge, this is the first work to cryptographically attest a virtualized PLC runtime via a trusted platform module.
- To the authors' best knowledge, this is the first work to develop a virtualized PLC environment generation approach which leverages existing system hardware and software artifacts to streamline a backfit deployment.
- To the authors' best knowledge, this is the first work to perform automated security orchestration in response to a PLC's failure to cryptographically attest.
- The proposed approach was implemented and evaluated through experimentation on a physical ICS power distribution testbed environment.

4.2 Related Work

The OT domain's hard and soft real-time performance requirements clash with virtualization's added layers of software complexity. As additional software processes are introduced to support PLC virtualization, the ability to guarantee real-time control loop

performance can be reduced. A previous paper addresses this problem by utilizing a real-time optimized environment and highlighting virtual PLC feasibility [43]. While the previous paper explores the feasibility of a virtualized PLC, this research explores the trust and resiliency functionality enabled by this proven virtual PLC feasibility within an approach that can conform to existing systems. Moreover, another paper explores the feasibility of cloud-based virtual PLC control with experimentation showing feasibility for soft real-time systems [44]. However, that paper only provided performance feasibility and this research's cryptographic attestation can be applied to address the trust concerns associated with cloud utilization.

Previous papers have proven attestation feasibility using the physics of the control process [45] [46]. This approach's PLC attestation technique does not observe the physical system state and could be implemented in combination with these other attestation methods. Additionally, this research utilizes security orchestration and SDN in combination with the attestation outcome of a virtualized PLC.

Our previous research defined and implemented an automated resiliency approach employing containerized PLCs [1]. While this previous research containerized its PLC runtime, this chapter's approach utilizes PLC virtualization to achieve additional security isolation. Additionally, the previous research acknowledges an approach shortcoming surrounding the manual generation of containerized PLC logic [1]. This research shortcoming is addressed in this chapter through the approach's ability to leverage existing software artifacts for automated virtual PLC generation. Lastly, the previous chapter's research states that the cryptographic attestation and subsequent response actions should be researched to guarantee system integrity

[1]. This research avenue is pursued through this chapter's cryptographically attested virtual PLC configuration.

4.3 Trust and Resilience: A Virtualization Approach

4.3.1 PLC Virtualization

When virtualizing a PLC, we aimed to satisfy several criteria. Primarily, that the virtualized PLC would mirror the physical PLC's control behavior. Secondly, that the virtual PLC could utilize the existing hardware and wiring installation to achieve system control. Lastly, that the process to virtualize the physical PLC could be automated and be capable of ingesting existing system software artifacts. These criteria assure virtual PLCs will function properly and minimize the cost and effort associated with their adoption.

To virtualize a PLC while conforming to these criteria, we defined an architecture which leveraged a software variant of the current PLC's runtime. In this research, we utilized Allen Bradley's SoftLogix 5800 version 23 PLC runtime. While we utilized an Allen Bradley product in this research due to their substantial North American market share, it is worth noting that most major PLC vendors offer a similar software-based PLC product, meaning this approach would translate to other PLC vendors and ecosystems. By virtualizing this software PLC, we inherit the ability to seamlessly interface with the existing PLC's ecosystem, to include software and hardware. As a result, this virtualized software PLC runtime required minimal effort to ingest and execute the logical artifacts of an existing PLC and could interface with its installed hardware endpoints, meeting the second and third defined criteria. Moreover, due to the ease of preexisting logic utilization, no logic reprogramming had to be performed. This execution of identical logic ensures that the virtual PLC's control loop would be identical, resulting in identical

process execution and conforming to the first defined criteria. Ultimately, this portion of the approach's outcome is the implementation of a logically equivalent virtualized software PLC which utilizes existing input and output hardware and wiring to drive the control system all the while being enhanced with virtualization's adaptability to threats, isolation protections, and lower costs.

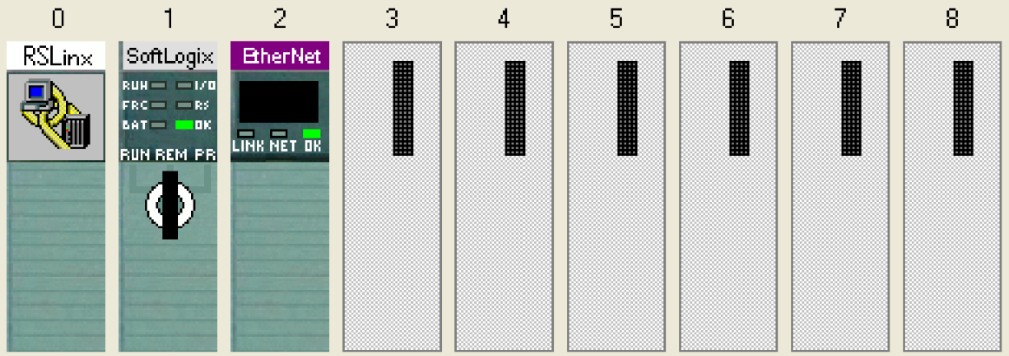


Figure 4-1 A SoftLogix PLC Chassis Represented in Software

4.3.2 Trusted Platform Module Remote Attestation

An additional benefit of virtualization is the ability to implement various security and trust mechanisms to the host and runtime, which could be unavailable to a proprietary hardware PLC solution. Due to the potentially critical processes PLCs control, trusting these devices is paramount to having confidence in process continuity and outcome. Additionally, due to this approach's reliance on virtualization for enhanced resilience, this can introduce new vulnerabilities unique to the IT domain. Therefore, a method to affirm the trust of virtual PLCs was researched. To achieve this end, we executed our virtual PLC on a host which had a trusted platform module (TPM) version 2.0. TPMs leverage cryptographic keys generated during the manufacturing process to perform remote attestation, a technique which ensures the legitimacy

of a networked device and its software. This research utilized a hardware-based TPM, a contrast to software based TPM solutions observed in prior research [47] [48].

In remote attestation, there exists a challenger host system which attempts to verify the internal state of another attestator system. The end goal of the attestation is to allow the attestator to generate a signed TPM quote that proves to the challenger that the internal state of the system matches the expected state [49]. The virtual PLC testbed used in this effort implemented remote attestation where the challenger is a Linux orchestration server and the attestator is the Intel Next Unit of Computing (NUC) that hosts the virtual PLC.

The first step in the attestation process is a one-time device registration. At this step, the challenger sends a request to register with the attestator, and the attestator generates an endorsement key (EK) along with an attestation identity key (AIK) pair. The EK is an asymmetric key that is unique to the TPM on the device. The AIK is generated by the TPM and is signed by the EK. To ensure the challenger is not registering with an imposter, a trusted third-party can perform certificate validation of the attestator. The next step in the registration is to compute the reference hash for the challenger. The reference hash is produced using the platform configuration registers (PCR) that are within the TPM. These registers can only be modified using a hash extension, which will overwrite any previous existing values. Since extension is the only path to overwrite values, the PCR hash values will reflect all the history of the hash extensions [50]. The attestator is sent the specific PCR to use for the hash computation. One portion of the registers is left untouched and is used for identifying the TPM hardware; the other set of registers is extended with the hash of a file that reflects what logic is being used by the virtual PLC. The file reflecting the logic is a proprietary Allen Bradley project file with the extension ‘acd’. The acd file

being used is contained in the SoftLogix program files and is labeled ‘slot02.acd’. Slot 2 in our SoftLogix program is the central processing unit (CPU) of the virtual PLC and the acd file in the file path location corresponds to the specific logic with which the CPU is currently programmed. Once the hash is computed, the AIK public key and hash are sent to the challenger, thus finishing the registration process.

With registration complete, the challenger is now ready to verify the internal state of the attestator at any time. To do so, the challenger will send a request to the attestator to read the PCR registers from which the computed hash was derived as well as a generated random nonce. The attestator computes the hash of the current acd file for Slot 02 and overwrites that hash into the PCR corresponding to the device’s software. With the PCR up to date, the attestator generates a TPM quote for the challenger using the PCR values and signs it with the private AIK. When the challenger receives the TPM quote, it first checks that the nonce and quote signature are valid. Lastly, it verifies that the PCR values received match the computed hash in the registration process. If any of the mentioned checks fail, it indicates that the internal state of the system has been modified. If the checks pass, then the internal state reflects what is expected, and the attestation has successfully passed. A diagram of the attestation process can be seen in Figure 4-2 below.

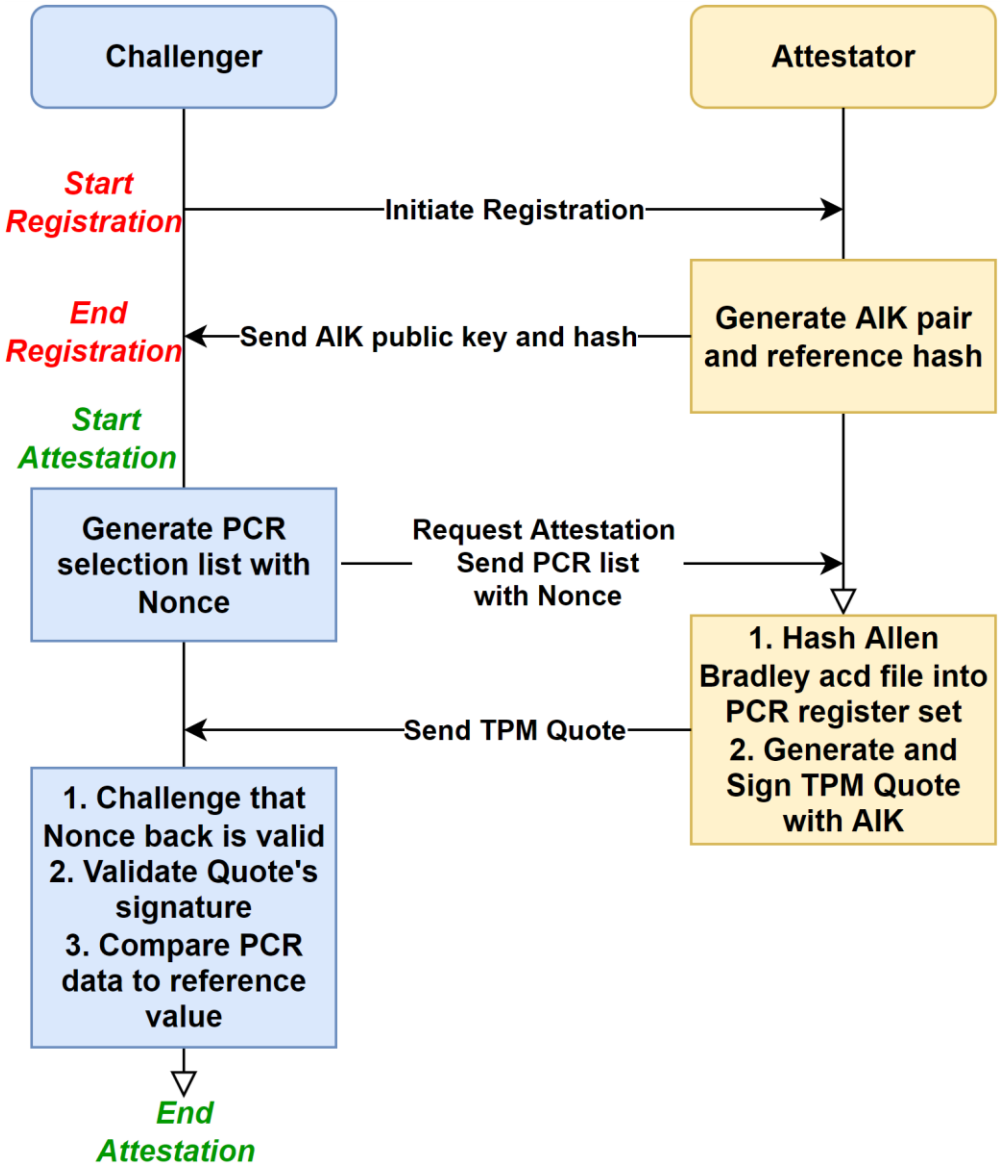


Figure 4-2 The Attestation Process

The result of the virtual PLC's attestation check was provided to our Security Information and Event Management (SIEM) which aggregates various system indicators and facilitates mitigation responses. If the SIEM reported that a virtual PLC failed to cryptographically attest its trustworthiness, it was not given control of the physical system and specific response actions could be taken to mitigate the situation.

4.4 Experimentation

The following subsections describe the power distribution experimentation environment and details the approach's experiments.

4.4.1 The Experimentation Environment

The experimentation occurred on a power distribution testbed which accepted user input via an HMI to open and close contactors and manipulate the flow of power to a target load. The system used an Allen Bradley 1756 ControlLogix PLC chassis for control. An Allen Bradley system was selected because they represent a large portion of PLC market share, making the results of the experimentation immediately applicable to thousands of existing systems due to the approach's backfit considerations [51]. Figure 4-3 below displays a diagram of the testbed environment.

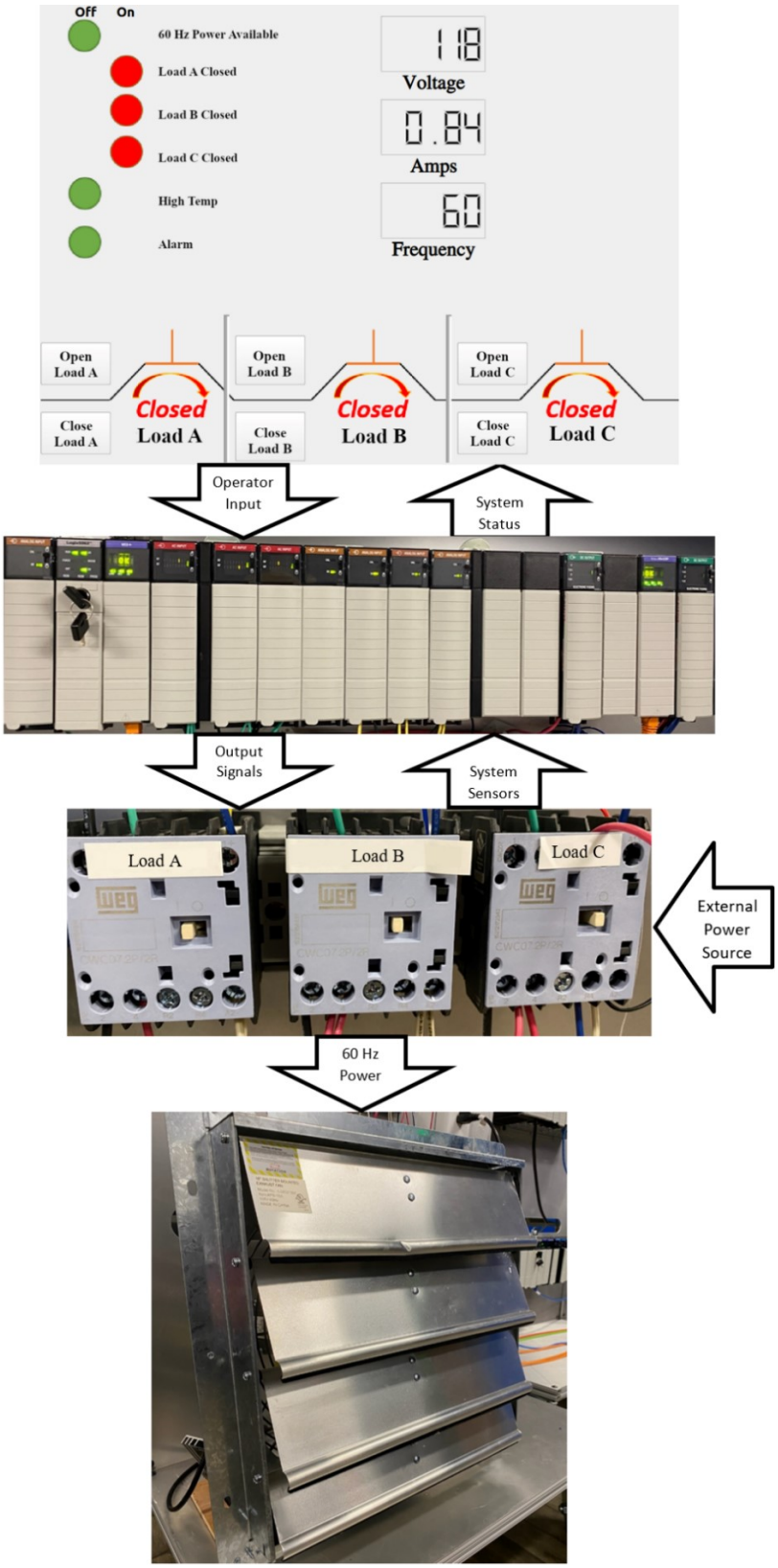


Figure 4-3 The Experiment's Power Distribution System Testbed

We began virtual PLC creation by launching our SoftLogix 5800 PLC runtime in a Windows 10 virtual machine (VM). Windows was utilized to satisfy the SoftLogix operating system (OS) requirements. Within Windows, the SoftLogix processes were set with real-time process priority, mitigating interruptions caused by other OS processes. This virtual machine was hosted on an Intel NUC running an Ubuntu 20.04 OS. Using Allen Bradley's proprietary software Studio 5000, we retrieved the logic and configuration from the existing physical PLC. Any hardware references in the original logic and configuration were changed to the virtual PLC instance and remote I/O. Nothing precludes this reconfiguration process from being completely automated. Lastly, despite this experiment's reliance on a proprietary PLC runtime, many PLC vendors offer comparable runtime products, and this approach could support a runtime substitution to a preferred vendor or open-source runtime solution [52]. The PLC runtime was virtualized using VMware ESXi and executed as a high-priority real-time OS process. Figure 4-4 below shows the virtual PLC host.

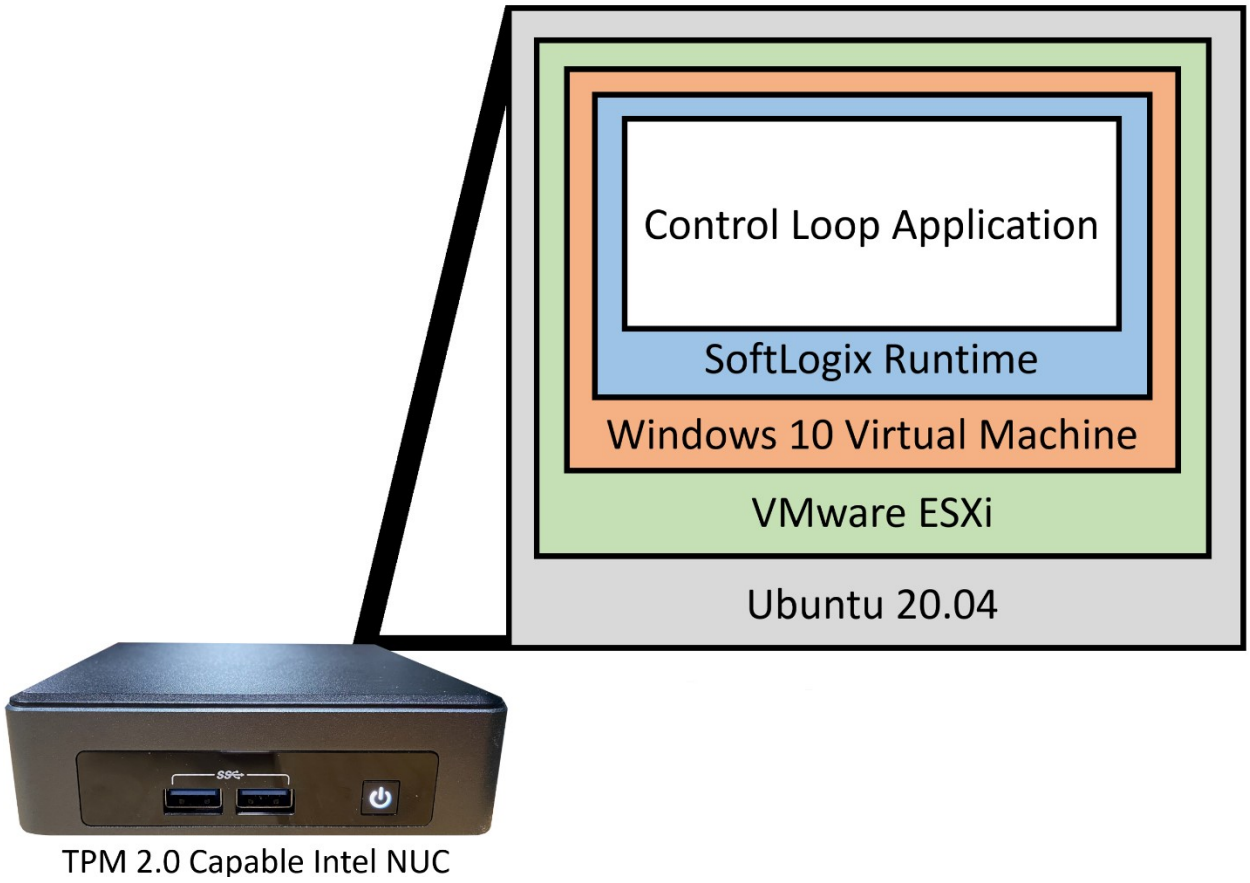


Figure 4-4 The Virtual PLC Host Architecture

The Intel NUC hosting the virtual PLC had TPM 2.0 hardware which was used to validate the Ubuntu 20.04 host OS as well as the virtual PLC's logic file. A snapshot of the configured and attested virtual PLC state was taken for later use during experimentation.

Supplemental to the trusted virtualized PLC is a security environment which enables automated alerting, analysis, and response actions. This environment consists of a rule-based Intrusion Detection System (IDS), an SDN controller and switch, and a Security Orchestration, Automation, and Response (SOAR) tool. The IDS was configured with a ruleset which could detect and alert on the modification of PLC logic. The SDN controller was programmed to redirect network traffic from the physical PLC to the virtual one via OpenFlow commands sent to the SDN

switch. The SOAR tool was configured with the automated actions to ingest IDS alerts, request the status of the virtual PLC's attestation, interface with the SDN controller to enact SDN-based network modifications, and prompt an operator to perform a manual action. Figure 4-5 below showcases this security environment.

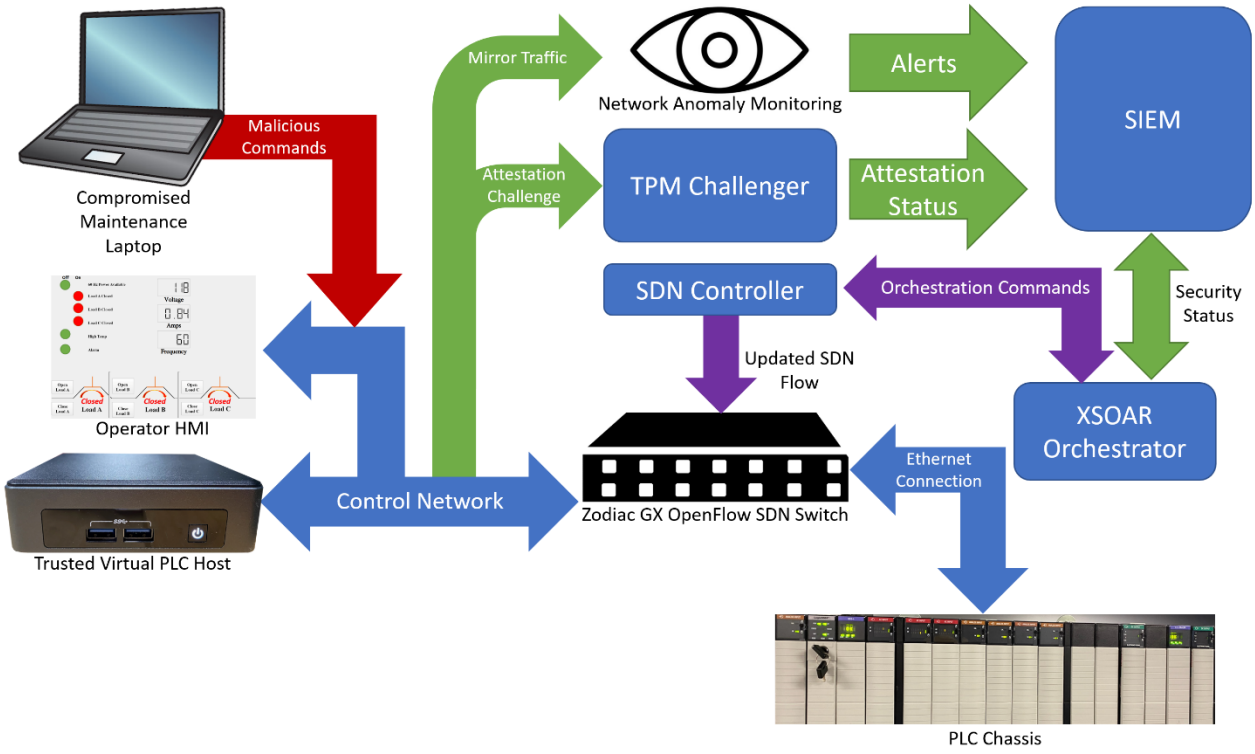


Figure 4-5 The Control System Network's Automated Security Implementation

4.4.2 The Virtual PLC Resilience Experiment

Experimentation began by confirming system functionality while under control of the virtual PLC. Initially, the virtual PLC failed to properly control the system. However, it was quickly discovered that the virtual PLC's logical variables were all initialized to zero, as it did not properly inherit the default variable values of the physical PLC. This was quickly corrected and expected system functionality was achieved under the virtual PLC's control.

With confirmation that the virtual PLC could control the system if needed, an attacker model was defined. The model leverages trusted access to reprogram a PLC, resulting in reduced system functionality. Specifically, any loads powered on would lose power after ten seconds. Additionally, without proper cyber situational awareness, the incident might be investigated as a hardware failure, resulting in system assets remaining unpowered for extended periods of time. This model's deployment of a trusted but malicious configuration represents a witting or unwitting insider threat, the compromise of a trusted device, or supply chain compromise.

The attacker model was exercised in three scenarios. The first scenario executed the attacker model against the system's physical PLC with no resiliency enhancements. The attacker modified the PLC's programming which resulted in an expected degradation of system behavior. Utilization of network situational awareness tools enabled a timely identification of the rogue machine and diagnosis of anomalous system behavior. Even so, the attacker model's configuration change continued to persist until manual reconfiguration could occur. Manual reconfiguration can take hours to weeks depending on a system's complexity and scale, the damage caused, and process sensitivity. Live duplicate systems are a common redundancy method but are still susceptible to the same attack due to the dependence on identical hardware and software.

In the second scenario, the physical PLC was compromised again; however, the network situational awareness tools enabled automated system recovery leveraging the trusted virtual PLC approach. This contrasts with scenario one, where utilizing this same network data only resulted in threat identification and system behavior diagnosis. The attacker's actions triggered a SOAR workflow which requested the virtual PLC's attestation status. Given that the virtual PLC's configuration was untouched, it passed the attestation check and was given responsibility of the

system's control loop via the SOAR's SDN orchestration. The SDN network reconfiguration took place after the operator was prompted to remove the infected PLC's CPU from the system. This CPU removal further purges the system of malicious artifacts and ensures there is no conflict over the I/O with the remaining PLC chassis. These corrective actions took seconds and resulted in expected control system behavior. Figure 4-6 shows the portion of the orchestration playbook which receives the passing attestation check and prompts follow-on actions. Additional details of the complete orchestration playbook are provided in Appendix A.

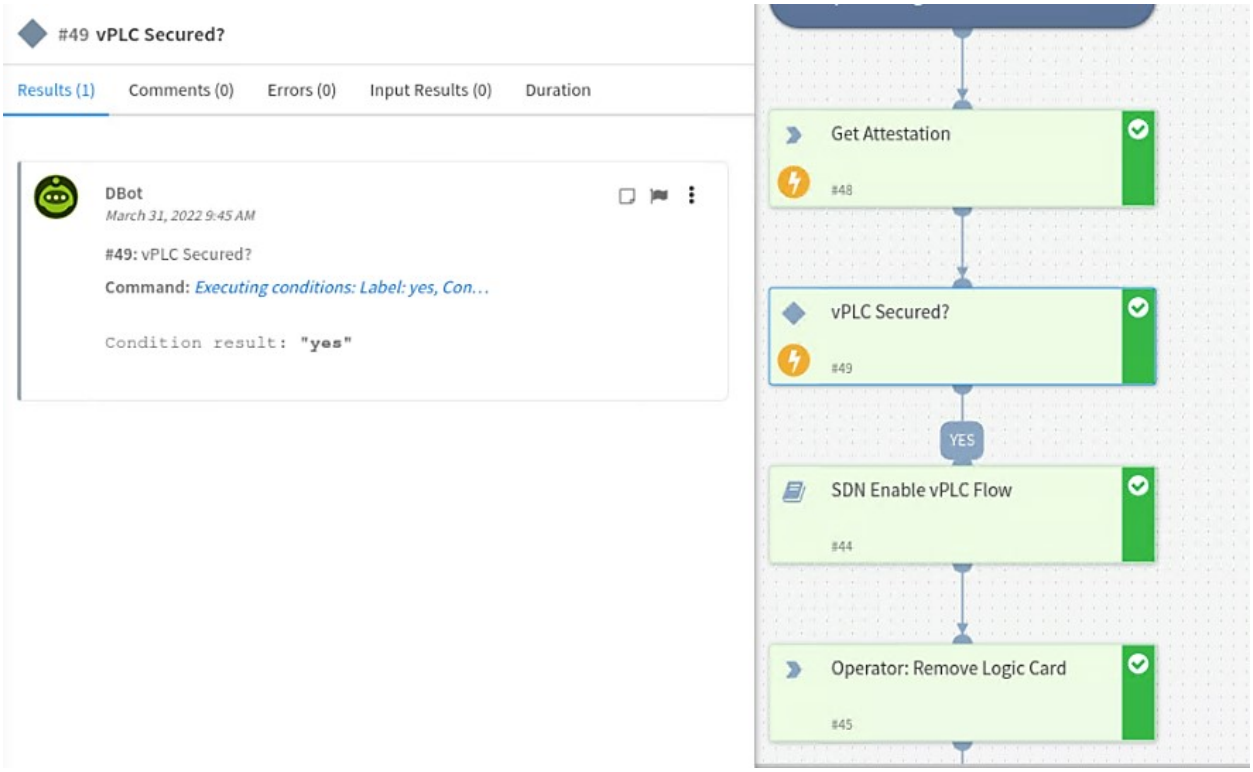


Figure 4-6 An Orchestration Playbook Checking the Attestation Status of the Virtual PLC

The third attack scenario executed the attacker model against both physical and trusted virtual PLCs. Due to the trusted status of the compromised maintenance laptop, a malicious configuration was accepted by both devices. When the SOAR executed its workflow for transitioning system control from physical PLC to virtual PLC, the SOAR workflow commanded an

attestation check of the virtual PLC. The virtual PLC failed to attest due to the discrepancy between the expected configuration and currently loaded malicious configuration. This failure to attest resulted in a complete loss of trust and confidence in the virtual PLC's configuration and the SOAR's workflow diverged to compensate. The SOAR then made an operator recommendation to revert to the snapshot of the virtual PLC in a known good configuration. While this action was given to an operator, it could easily be automated via scripting or Application Programming Interface (API) calls. After reverting to the snapshot state, the device passed the attestation check and was given physical system control via the SOAR's SDN orchestration after the PLC chassis' CPU removal. This process took one minute to complete, with the result being the semi-automated transition to a trusted system. The system performed as expected and all operator input produced the appropriate output.

In scenarios two and three, the speed of the automated process is limited by the operator's removal of the infected CPU. This could be easily remedied by moving the physical CPU to a separate PLC chassis and using the remote I/O protocol to control the I/O. With both virtual and physical PLC CPUs using remote I/O through the SDN switch, the process could be made completely automated. This requires some minor reconfiguration of the existing PLC system, unlike the tested method which is more immediately applicable to existing systems. Figure 4-7 below showcases this automated system architecture.

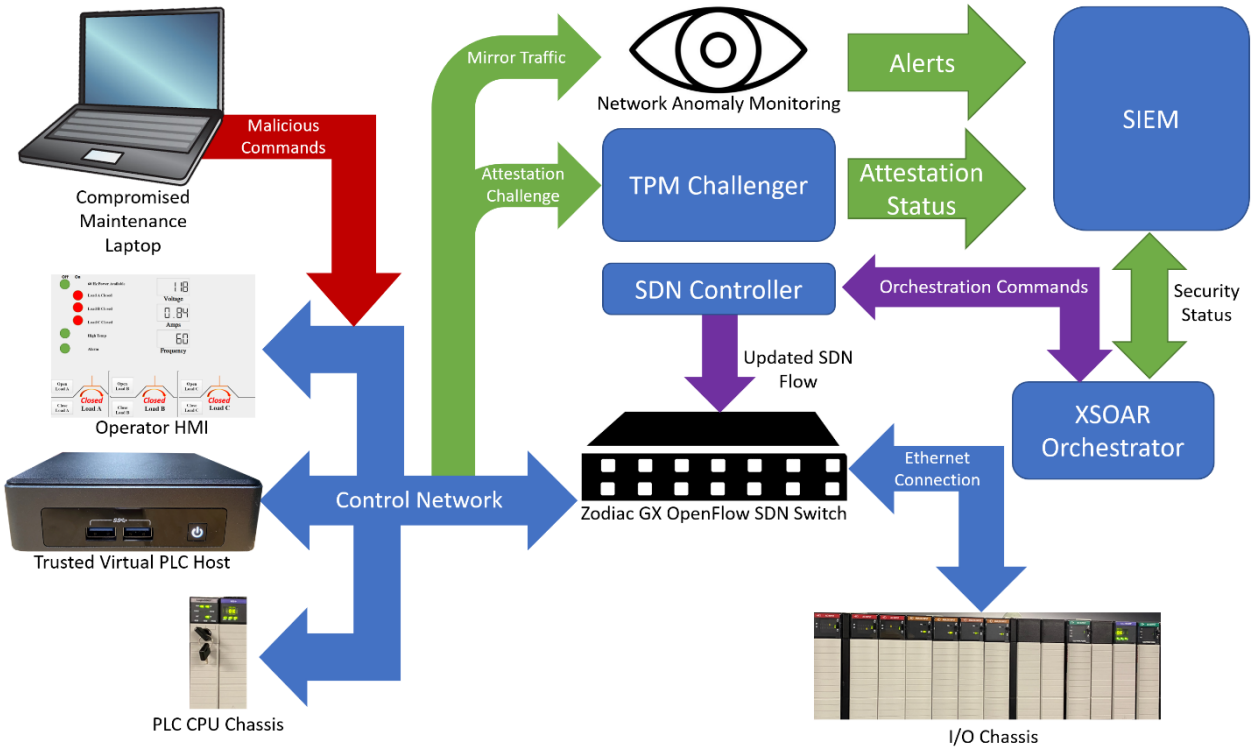


Figure 4-7 The Automated Security Architecture with Seamless Virtual PLC Failover Support

4.5 Experimentation Results

Similar to the previous water treatment and water chiller experiments performed using containerized PLCs, four questions were used to establish and verify system behavior. Throughout experimentation, the system was stimulated and observed to validate the system's behavior when under the control of the virtual PLC. This behavior was validated against a baseline of observed system behavior prior to experimentation. Table 4-1 below displays the observations.

Table 4-1 Observed System Characteristics During Experimentation Scenarios

Experimentation Scenario	**System Characteristic Observations (Yes/No)**			
	Are reported sensor values	Is the physical process	Are system network communications	Does the operator interface

	reflecting expected values?	behaving as expected?	functioning as expected?	exhibit nominal behavior?
Scenario One: Baseline Adversary & No Virtual PLC Utilization	Yes	No	Yes	No
Scenario Two: Baseline Adversary & Trusted Virtual PLC Utilization	Yes	Yes	Yes	Yes
Scenario Three: Advanced Adversary & Trusted Virtual PLC Utilization	Yes	Yes	Yes	Yes

Scenario one shows the attacker model's effectiveness at maliciously affecting the control process when the approach's resiliency benefits were not present. Following the orchestrated transition to the virtualized PLC in scenarios two and three, the system exhibited identical behavior when compared to the expected behavior established by the baseline. Comparing the outcomes between scenarios one and two, showcases the automated resiliency benefits the virtual PLC approach provides. This is evidenced by continuity of the control process, and by the execution speed advantages associated with automated recovery compared to manual recovery. Furthermore, comparing the outcomes between scenarios two and three accentuates the need for system trust while also demonstrating the virtual PLC's flexibility. This is evidenced by the control system's process continuity despite the initial virtual PLC compromise executed by the trusted host.

4.6 Conclusions

The virtual PLC's initial misconfiguration due to a lack of variable initialization values highlights the need for automated validation of virtual PLC generation. This validation could be performed using a simulation environment for stimulation of virtual PLC execution or an automated comparison of the cryptographic hashes between physical and virtual PLC programming. Additionally, because the proprietary PLC's software runtime was never meant to be virtualized, its licensing model could be more cost effective at scale. This is due to the expectation that a physical machine would be dedicated to a single software instance as opposed to a singular physical machine potentially running multiple virtual instances. This is a shortcoming which is subject to change with market demands as this approach's feasibility continues to be proven and successful technology pilots occur, potentially leading to alternative site-wide licensing models.

The successful demonstration of the virtualization approach throughout the experimentation scenarios highlights its effectiveness for ensuring process continuity during a cyberattack. It is also worth noting that the virtual PLC's added flexibility facilitates the implementation of additional security mechanisms that a proprietary PLC system may not support, reducing the likelihood of compromise and introducing new host-based data sources. This approach's trusted flexibility, paired with its means to augment existing and proprietary environments enables these systems to take dynamic actions in support of their processes and could play a pivotal role in securing both existing and future systems all the while introducing the cost-savings and high-availability benefits of virtualization.

4.7 Future Work

Automated validation of generated virtual PLCs should be investigated to ensure attested configurations conforms to the demands of the physical processes. Data fusion which correlates the virtual PLC's attestation with additional data sources should be investigated to define a holistic solution which provides trust indicators for the system's processes. For example, associating physical sensor data with an attestation failure could quantify the impact of an unauthorized change on the physical process. This data fusion would also result in more effective automated recovery methods due to the included understanding of the physical system's state. Moreover, as additional data sources and indicators are ingested, an autonomic decision engine could orchestrate the process continuity actions in cooperation with the manually developed rulesets deployed in this chapter. This would enable this approach to mitigate anomalous events which fall outside of the defined ruleset's parameters.

This chapter references research which validates the approach's computational speed in relation to soft real-time applications. However, additional testing and tuning should be performed to understand and maximize the implementation's real-world performance. Additionally, due to the diversity of systems within the OT domain, applying this approach to non-PLC control system devices could further expand the approach's relevance with candidate examples being building automation controllers and purpose-built power grid equipment. Moreover, this approach's added processing resilience at the OT device-level has the potential to shift the attacker's targeting priority to the firmware of the actuators, sensors, and I/O to gain a foothold below this approach's current applicability, research to establish and maintain trust at this low level establishes a more holistic OT cybersecurity solution. Lastly, technology pilots

which utilize environments with varying scales and operational requirements should be performed to explore the approach's breadth of applicability.

5 Summary

In this dissertation, we have explored novel approaches to enhancing ICS resilience. We researched and examined an exemplar ICS cyber threat and postulated future domain threats [2]. The results of this examination, combined with the understanding of current domain research guided and benchmarked subsequent research. We researched an approach which applied containerization technology and I/O multiplexing to multiple ICS laboratory environments which enhanced their ability to recover from cyber compromise and ensure system continuity [1]. Lastly, we researched an approach which enhances the resilience, adoptability, timeliness, and trust of the containerized approach by incorporating virtualization, SDN, orchestration, cryptographic assuredness, and seamless backfit integration [3]. All research presented in this dissertation was published in academic peer-reviewed settings.

Consequential to researching these approaches, resulting technologies were developed which have since been piloted in critical control system environments [6]. Additionally, the approaches' accurate hardware abstraction has been utilized to generate large-scale, high-fidelity environments and data which currently support research validation in the areas of cyber response training effectiveness, artificial intelligence anomaly detection algorithms, and information influences on attacker behavior. Ultimately, this dissertation's research addresses critical domain challenges and applies resilience to many systems with unmatched societal importance while serving as an enabler for future research for years to come.

6 References

1. J. Cervini, A. Rubin and L. Watkins, "A Containerization-Based Backfit Approach for Industrial Control System Resiliency," 2021 IEEE Security and Privacy Workshops (SPW), pp. 246-252, 2021, doi: 10.1109/SPW53761.2021.00043.

2. J. Cervini, A. Rubin and L. Watkins, "Don't Drink the Cyber: Extrapolating the Possibilities of Oldsmar's Water Treatment Cyberattack," Seventeenth International Conference on Cyber Warfare and Security (ICCWS), pp. 246-252, 2022, doi: 10.34190/iccws.17.1.29.

3. J. Cervini, D. Muller, A. Beall, J. Maurio, A. Rubin and L. Watkins, "A Backfit Approach for Trusted Virtualization-Based Programmable Logic Controller Resilience," Sixteenth International Conference on Critical Infrastructure Protection XVI. ICCIP 2022.

4. Cybersecurity and Infrastructure Security Agency, "Critical Infrastructure Sectors," [Online]. Available: https://www.cisa.gov/critical-infrastructure-sectors.

5. National Institute of Standards in Technology, "Framework for Improving Critical Infrastructure Cybersecurity," pp. 23, 16 April 2018. [Online]. Available: https://nvlpubs.nist.gov/nistpubs/CSWP/NIST.CSWP.04162018.pdf.

6. The United States Coast Guard Research and Development Center, "Demonstration of Virtualized Programmable Logic Controllers for Response and Recovery of Operational Technology Systems," System for Award Management, April 2019. [Online]. Available: https://sam.gov/opp/f00d58cd5cc56fd2c66262e48b9a1024/view.

7. M. Krotofil and D. Gollmann, "Industrial control systems security: What is happening?,"11th IEEE International Conference on Industrial Informatics (INDIN), pp. 670-675, 2013.

8. Hobbs, Allegra., "The colonial pipeline hack: Exposing vulnerabilities in US cybersecurity," SAGE Publications: SAGE Business Cases Originals, 2021.

9. Gottlieb, Otto., "Tips for Allen-Bradley Control Panel Design," DMC Inc., 2020, [Online]. Available: https://www.dmcinfo.com/latest-thinking/blog/id/10116/tips-for-allen-bradley-control-panel-design.

10. Connell Industries, "PLC and HMI Retrofit Systems," [Online]. Available: https://connell-ind.com/PLC_and_HMI_Retrofits.

11. Malena Carollo, Jack Evans, "Oldsmar's water supply attack is a warning, experts say. It could've been worse." February 2021. [Online]. Available: https://www.tampabay.com/news/pinellas/2021/02/10/oldsmars-water-supply-attack-is-a-warning-experts-say-it-couldve-been-worse.

12. Alexander, O., Belisle, M. and Steele, J., “MITRE ATT&CK® for Industrial Control Systems: Design and Philosophy,” 2020.

13. Burks, D., “Security Onion,” 2012.

14. Challener, D.C., Kruus, P.S., Fink, R.A. and Farlow, J.F., Johns Hopkins University, 2018. “Apparatus and method for preventing access by malware to locally backed up data,” U.S. Patent 10,049,215.

15. Cybersecurity and Infrastructure Security Agency (CISA), “Malcolm,” 2021. [Online]. Available: https://github.com/cisagov/Malcolm.

16. City of Oldsmar, “City of Oldsmar Fiscal Year 2021/2022 Annual Budget,” pp. 111, 2021.

17. City of Oldsmar: Water Division, “Hours,” 2021.

18. Department of Homeland Security, “Water and Wastewater Systems Sector-Specific Plan,” 2015.

19. Heenan, R. and Moradpoor, N., “Introduction to security onion.” The First Post Graduate Cyber Security Symposium, 2016.

20. Payne, B. and Mienie, E., “Multiple-Extortion Ransomware: The Case for Active Cyber Threat Intelligence,” ECCWS 2021 20th European Conference on Cyber Warfare and Security, pp. 331, 2021.

21. Petcu A. G., “The Curious Case of the Baltimore Ransomware Attack: What You Need to Know," 2020.

22. D. Peterson, "Quickdraw: Generating Security Log Events for Legacy SCADA and Control System Devices," 2009 Cybersecurity Applications & Technology Conference for Homeland Security, pp. 227-229, 2009.

23. Pinellas County Sheriff’s Office, “Treatment Plant Intrusion Press Conference,” 2021.

24. Reeder, J.R. and Hall, C.T., “Cybersecurity’s Pearl Harbor Moment: Lessons Learned from the Colonial Pipeline Ransomware Attack,” 2021.

25. RockNSM, “What is ROCK,” 2019.

26. ThreatLocker Incorporated, “Protecting Water Infrastructure Against Cyberattacks,” 2021.

27. Veritas Technologies LLC, “The 2020 Ransomware Resiliency Report,” 2020.

28. R. Langner, "Stuxnet: dissecting a cyberwarfare weapon," in IEEE Security & Privacy, vol. 9, no. 3, pp. 49-51, May-June 2011.

29. P. González-Nalda, I. Etxeberria-Agiriano, I. Calvo, and M. C. Otero, “A modular CPS architecture design based on ROS and Docker,” Int J Interact Des Manuf 11, pp. 949–955, 2017.

30. H. Lin, J. Zhuang, Y. Hu, and H. Zhou, “DefRec: Establishing Physical Function Virtualization to Disrupt Reconnaissance of Power Grids’ Cyber-Physical Infrastructures,” NDSS Symposium, February 2020.

31. M. Sollfrank, F. Loch, S. Denteneer, and B. Vogel-Heuser, "Evaluating Docker for Lightweight Virtualization of Distributed and Time-Sensitive Applications in Industrial Automation," IEEE Transactions on Industrial Informatics, 2020.

32. T. Alves and T. Morris, “OpenPLC: An IEC 61131-3 Compliant Open Source Industrial Controller for Cyber Security Research,” Computers & Security, 78, pp. 364-379, 2018.

33. W. Peters, “Integrated Adaptive Cyber Defense: Integration Spiral Results” Workshop on Automated Decision Making for Active Cyber Defense, 2015.

34. WidgetLords Electronics, “Pi-SPi-2A0 Raspberry Pi Analog Output (mA + VDC) Interface,” [Online]. Available: https://widgetlords.com/collections/4-20-ma-products/products/pi-spi-2a0-raspberry-pi-analog-output-ma-vdc-interface.

35. ProSoft Technology, “Modbus TCP/IP Client/Server Enhanced Network Interface Module for ControlLogix,” [Online]. Available: https://www.prosoft-technology.com/Products/Rockwell-Automation-In-chassis/Platform/ControlLogix/Modbus-TCP-IP-Client-Server-Enhanced-Network-Interface-Module-for-ControlLogix.

36. UniPi Technology, "UniPi Neuron L513," [Online]. Available: https://www.unipi.technology/unipi-neuron-l513-p106.

37. LabJack Corporation, “LabJack T7,” [Online]. Available: https://labjack.com/products/t7.

38. Z. Birnbaum, M. Davis, S. Salman, J. Cervini, L. Watkins, S. Yamajala, and S. Paul, “Cyber-Resilient SCADA Systems via Secure State Restoration,” In: Staggs J., Shenoi S. (eds) Critical Infrastructure Protection XIV. ICCIP 2020. IFIP Advances in Information and Communication Technology, vol 596. Springer, Cham. https://doi.org/10.1007/978-3-030-62840-6_9

39. G. De La Torre Parra, P. Rad, and K. Raymond Choo, “Implementation of Deep Packet Inspection in Smart Grids and Industrial Internet of Things: Challenges and Opportunities,” Journal of Network and Computer Applications, vol. 135, pp. 23-46, 2019.

40. D. Formby, P. Srinivasan, A. Leonard, J. Rogers, and R. Beyah, "Who's in Control of Your Control System? Device Fingerprinting for Cyber-Physical Systems," NDSS, 2016.

41. Rockwell Automation Incorporated, "ControlLogix System User Manual," pg. 150, October 2017. [Online]. Available: https://literature.rockwellautomation.com/idc/groups/literature/documents/um/1756-um001_-en-p.pdf.

42. Siemens, "S7-1200 System Manual," pg. 81, April 2012. [Online]. Available: https://cache.industry.siemens.com/dl/files/465/36932465/att_106119/v1/s71200_system_manual_en-US_en-US.pdf.

43. T. Cruz, P. Simões and E. Monteiro, "Virtualizing Programmable Logic Controllers: Toward a Convergent Approach," IEEE Embedded Systems Letters, vol. 8, no. 4, pp. 69-72, Dec. 2016.

44. O. Givehchi, J. Imtiaz, H. Trsek and J. Jasperneite, "Control-as-a-service from the cloud: A case study for using virtualized PLCs," 10th IEEE Workshop on Factory Communication Systems (WFCS 2014), pp. 1-4, 2014.

45. M. Salehi and S. Bayat-Sarmadi, "PLCDefender: Improving Remote Attestation Techniques for PLCs Using Physical Model," in IEEE Internet of Things Journal, vol. 8, no. 9, pp. 7372-7379, May 2021.

46. H.R. Ghaeini, M. Chan, R. Bahmani, F. Brasser, L. Garcia, J. Zhou, A.R. Sadeghi, N. O. Tippenhauer and S. Zonouz, "PAtt: Physics-based Attestation of Control Systems," 22nd International Symposium on Research in Attacks, Intrusions and Defenses (RAID), pp. 165-180, 2019.

47. A. Seshadri, A. Perrig, L. van Doorn and P. Khosla, "SWATT: softWare-based attestation for embedded devices", IEEE Symposium on Security and Privacy, pp. 272-282, 2004.

48. F. Armknecht, A. R. Sadghi, S. Schulz and C. Wachsmann, "A security framework for the analysis and design of software attestation," ACM SIGSAC conference on Computer and communications security, pp. 1-12, 2013.

49. A. Francillon, Q. Nguyen, K. B. Rasmussen and G. Tsudik, "A minimalist approach to Remote Attestation," 2014 Design, Automation & Test in Europe Conference & Exhibition (DATE), pp. 1-6, 2014.

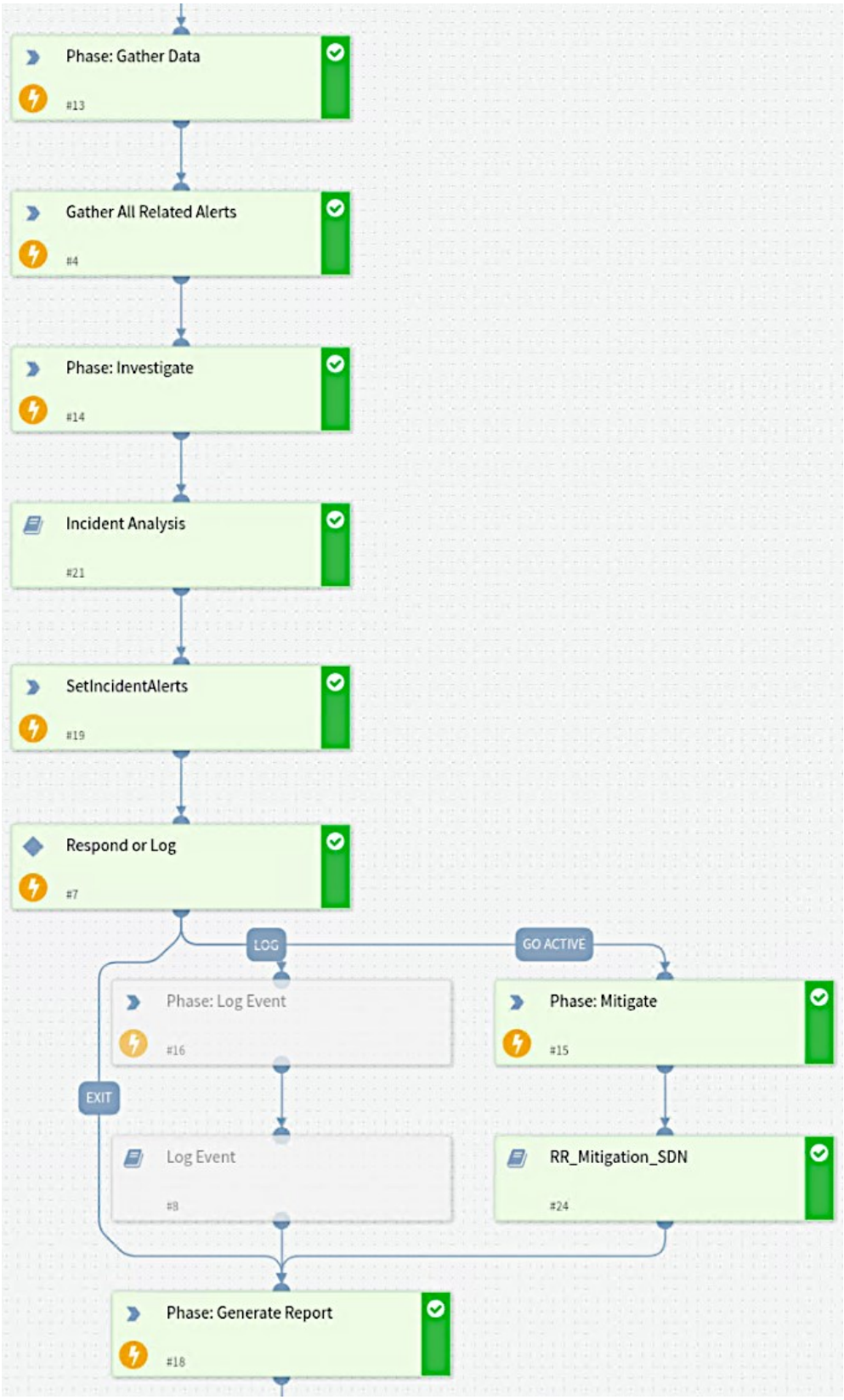
50. tpm2-software community, "Remote attestation," Dec. 18, 2019. [Online]. Available: https://tpm2-software.github.io/tpm2-tss/getting-started/2019/12/18/Remote-Attestation.html.

51. T. Dawson, “Who were the leading vendors of industrial controls in 2017?,” Nov. 2018, [Online]. Available: https://www.interactanalysis.com/who-were-the-leading-vendors-of-industrial-controls-in-2017.

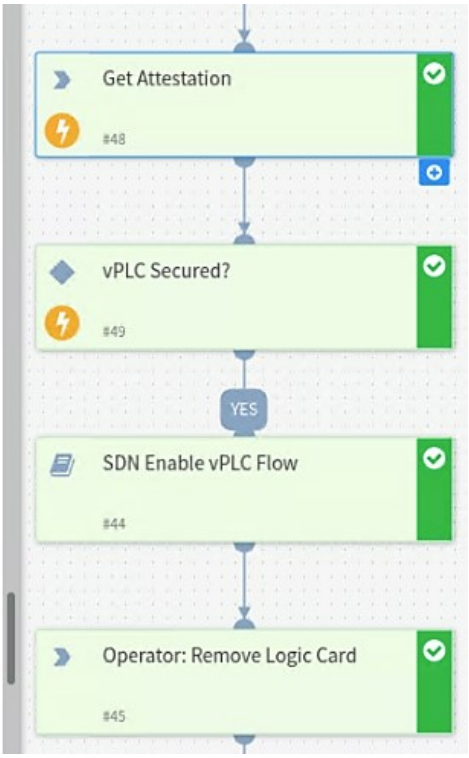
52. S. Fujita, K. Hata, A. Mochizuki, K. Sawada, S. Shin and S. Hosokawa, “OpenPLC based control system testbed for PLC whitelisting system,” Artificial Life and Robotics, vol. 26, pp. 149–154, 2021.

53. Riptide, “PyModbus – A Python Modbus Stack,” [Online]. Available: https://pymodbus.readthedocs.io/en/3.0.0/readme.html#summary.

Appendix A. Orchestration Playbook



The “RR_Mitigation_SDN” portion of the XSOAR playbook seen in the flow diagram above executes the subroutine playbook which checks the attestation status of the virtual PLC using a call to the Elastic SIEM, executes enacts the network modifying SDN flow, and prompts the operator to remove the logic card.



Appendix B. Hardware Interpreter Code Sample

Below is a simplified sample of the hardware interpreter querying the containerized PLC's state and setting an appropriate output pin:

```
#The line below opens the file which controls the relay output pin 2.01 using the SysFS driver.
relayOutput2_01 = open("/sys/devices/platform/unipi_plc/io_group1/ro_2_01/ro_value","w")

if multiplex: #Conditional which results in containerized PLC control if true.
  cPLC_Coils = cPLCClient.read_coils(0,8) #PyModbus read for the container PLC's first 8 values.
  if cPLC_Coils: #Check that the PyModbus read was successful
    if cPLC_Coils.bits[0]: #Check if the first container PLC value is true.
      relayOutput2_01.write("1") #Energize the pin if true.
    else:
      relayOutput2_01.write("0") #Deactivate the pin if false.
else: #Replicate the physical PLC's output.
  #The line below opens the file which is wired to the physical PLC's output using SysFS.
  inputPin1_01 = open("/sys/devices/platform/unipi_plc/io_group1/di_1_01/di_value","rt")
  if inputPin1_01.read().strip() == "1": #Conditional to check if the pin is energized.
    relayOutput2_01.write("1") #Energizes the relay output pin if the input pin is energized.
  else:
    relayOutput2_01.write("0") #Deactivate the relay output pin if the input pin is deactivated.

relayOutput2_01.close() #Closing the relayOutput2_01 file
```

Appendix C. Acronyms

µs	Microsecond
AIK	Attestation Identity Key
API	Application Programming Interface
BITW	Bump-In-The-Wire
CISA	Cybersecurity and Infrastructure Security Agency
cPLC	Containerized Programmable Logic Controller
CPU	Central Processing Unit
CPS	Cyber-Physical System
DENG	Doctor of Engineering
DHS	Department of Homeland Security
DPI	Deep Packet Inspection
EK	Endorsement Key
EST	Eastern Standard Time
HMI	Human Machine Interface
I/O	Input/Output
ICMP	Internet Control Message Protocol
ICS	Industrial Control System
IDS	Intrusion Detection System
IT	Information Technology
JHU	The Johns Hopkins University
JHU/APL	The Johns Hopkins University Applied Physics Laboratory
mA	Milliampere
MITM	Man-In-The-Middle
ms	Milliseconds
N/A	Not Applicable
NC	Normally Closed
NIST	National Institute of Standards in Technology
NUC	Next Unit of Computing

OS	Operating System
OT	Operational Technology
PCR	Platform Configuration Register
pH	Potential Hydrogen
PLC	Programmable Logic Controller
pps	Packets per Second
ppm	Parts per Million
SCADA	Supervisory Control and Data Acquisition
SDN	Software Defined Networking
SIEM	Security Information and Event Management
SMS	Short Message Service
SOAR	Security Orchestration, Automation, and Response
TPM	Trusted Platform Module
TTPs	Tactics Techniques and Procedures
V	Volts
VM	Virtual Machine
vPLC	Virtualized Programmable Logic Controller

Appendix C. Curriculum Vitae

James D. Cervini received his B.S. degree in Computer Science from the University of North Carolina Charlotte in 2016. Funded under the National Science Foundation Scholarship for Service program, he received his M.S. in Security Informatics from The Johns Hopkins University Information Security Institute in 2017 where his capstone project explored and analyzed the cybersecurity of wireless smart meter systems.

James has worked for The Johns Hopkins University Applied Physics Laboratory since 2017. He enrolled in the Doctor of Engineering program in the JHU Whiting School of Engineering in 2019. His research interests include cyber-physical system security, virtualization, fog computing, and penetration testing. His personal interests include building computers, classical cinema, performance automobiles, and traveling.