Towards Accurate and Reliable Industrial Intrusion Detection Systems Using Shadow Replicas

Kenechukwu A. Nwodo

Thesis submitted to the Faculty of the
Virginia Polytechnic Institute and State University
in partial fulfillment of the requirements for the degree of

Master of Science
in
Computer Engineering

Angelos Stavrou, Chair
Haining Wang
Paul K. Ampadu

May 10th, 2023
Arlington, Virginia

Keywords: Shadow Replica, Intrusion Detection, Supervisory Control and Data Acquisition

Towards Accurate and Reliable Industrial Intrusion Detection Systems Using Shadow Replicas

Kenechukwu A. Nwodo

(ABSTRACT)

Supervisory Control and Data Acquisition (SCADA) systems manage the operations of a plethora of safety-critical industrial control systems. Due to their sensitive nature, SCADA systems have been the target of adversaries employing a wide range of attacks. This thesis proposes an approach to protect SCADA systems against attacks that evade detection because of the lack of a comprehensive view of both application and network-layer responses. Specifically, we leverage multiple open-source Network Intrusion Detection Systems (NIDSs) paired with a SCADA shadow replica to provide both network and application threat detection. The shadow replica is augmented with a Finite State Machine (FSM) to compute the anticipated states of both the SCADA system and connected devices. Isolated from the operational network, it is protected from direct front-end attacks. When the SCADA system becomes compromised, even without an IDS alert, the replica can expose the attack and offer an operational failover. We implement a prototype of our system and evaluate it against locally executed attacks on commercial out-of-the-box devices and public IoT datasets. Results indicate that incorporating the shadow replica alongside NIDSs can enhance detection coverage in our evaluations.

Towards Accurate and Reliable Industrial Intrusion Detection Systems Using Shadow Replicas

Kenechukwu A. Nwodo

(GENERAL AUDIENCE ABSTRACT)

As the number of network-enabled industrial control devices and sensors increase, so does the importance of the central systems that control them, known as Supervisory Control and Data Acquisition (SCADA) systems. This interconnectedness, however, has led to a rise in cyberattacks attempting to breach these critical devices. This thesis proposes a novel approach to protect SCADA systems from threats that exploit specific network behaviors. We pair network monitoring tools such as Network Intrusion Detection Systems (NIDSs) with a shadow replica, which acts as a digital mirror of the system. To track the states of the system and protect the shadow replica from direct attacks, we utilize a mathematical model called Finite State Machines (FSMs). We designed and implemented this system by integrating commercial sensors with an open source SCADA controller. Our experiments show that using a shadow replica provides an additional layer of defense against these attacks.

Dedication

To My Family

Acknowledgments

First and foremost, Thank You God.

I would like to express my sincerest gratitude to my advisor, Dr. Angelos Stavrou, for his guidance and support throughout this academic journey. I also extend my appreciation to my co-advisor, Dr. Haining Wang, for his valuable advice and insights. I would like to thank my collaborator on this work, Mahsa Foruhandeh, as well as my lab mates.

I am deeply grateful to my parents and sister, whose mentorship, love, and support have made this achievement possible. I truly thank my family for being my role models.

Finally, I thank my friends, former professors, teachers, and colleagues who have all helped me reach this milestone.

Contents

List of Figures

List of Tables

List of Abbreviations

AP Access Point

BEMOSS Building Energy Management Open Source Software

DoS Denial of Service

FSM Finite-state Machine

HTTP Hypertext Transfer Protocol

HVAC Heating, Ventilation, and Air-conditioning

IDS Intrusion Detection System

IIDS Industrial Intrusion Detection System

IoT Internet of Things

MITM Man In The Middle

NIDS Network-based Intrusion Detection System

PLC Programmable Logic Controller

RTU Remote Transmission Unit

SCADA Supervisory Control and Data Acquisition

SYN Synchronize

TCP Transmission Control Protocol

Chapter 1

Introduction

Uninterrupted operation is paramount for cyber-physical systems, especially when it comes to life-critical systems, including healthcare and industrial control systems. For this reason, countries like the United States are releasing new cybersecurity strategies, highlighting the defense of critical infrastructure as a major necessity [24]. These infrastructures, oftentime, utilize supervisory control and data acquisition (SCADA) systems, which are highly distributed systems employed to control and monitor geographically dispersed assets. Such assets can be edge or core devices such as actuators, sensors, protection devices, or other smart devices. SCADA systems typically comprise multiple functional layers, and communication between the SCADA controller and assets may use protocols such as Wi-Fi, Bluetooth [64], Zigbee [36], among others.

Security threats to industrial systems include attack vectors targeting the network and application layers. Through code injections, adversaries can send malicious commands via network payloads to disrupt application-layer operations [43]. Following the methodology of remote execution attacks (e.g., the 2021 Log4j [47]), industrial control system devices can be compromised by rogue commands that alter their states. Existing industrial intrusion detection systems (IIDSs) aim to thwart such threats using protocol-specific signatures and behavioral models [58]. While effective for detecting denial of service (DoS) attacks, these signatures and models are not mature enough to stop zero-day exploits or code injection attacks [29, 40, 49]. As a result, when SCADA systems are targeted by such attacks, it

becomes difficult to determine whether the resulting behavior is legitimate.

To address this security gap, we propose an IIDS solution that integrates off-the-shelf open-source network IDSs (NIDSs) with a shadow replica. A shadow replica is a separate monitoring and control device located outside the SCADA system's operational network, capable of taking over control when necessary. This provides a fail-safe mechanism if an attack manages to disrupt operational flow. A key feature of our shadow replica is its finite state machine (FSM), which mathematically models and forecasts the SCADA system's future states. This enables the detection of malicious activity by identifying anomalous state transitions that violate expected system logic.

Our approach combines an FSM-based shadow replica and open-source NIDSs for IoT intrusion detection. Every packet processed by the original controller is mirrored to the shadow replica. Because the replica is isolated from the application layer front, attacks targeting that layer cannot reach it. By comparing the FSM-predicted state transitions of the shadow replica to the actual state changes in the original SCADA system, our framework detects malicious activity. When a novel attack is identified, the shadow replica flags it and uses the anomaly information to update the NIDS, maintaining uninterrupted service. Moreover, it can verify alerts raised by the NIDS itself.

For evaluation, we implement our architecture using the Building Energy Management Open Source Software (BEMOSS) platform [38], chosen for its configurability and compatibility with a wide range of devices, over alternatives like Metasys [4] and Desigo CC [3]. This setup enables the development and testing of a prototype for the proposed defense mechanism. During our evaluation, we account for potential sources of false positives and negatives, such as time drifts between the replica and NIDS, and mismatched signatures among NIDS components. These issues are mitigated through resyncing and retraining the system as needed, and employing two different signature-based NIDSs in addition to the replica, Suricata [53]

and Zeek [65], to expand detection coverage. We first experimented with detecting locally executed attacks, then conducted comprehensive evaluations with public IoT datasets, comparing NIDS recall values with the improved coverage achieved when the shadow replica is incorporated.

1.1 Key Contributions

This thesis makes the following contributions:

- Designed a tandem IIDS solution comprising two network-based intrusion detection systems connected to a shadow replica augmented with an FSM event parser.
- Implemented a prototype of the proposed tandem IIDS and evaluated its efficacy through cyberattacks on SCADA systems, demonstrating the ability to detect threats at the network and application layers.
- Benchmarked the solution on public IoT datasets using the shadow replica alongside NIDSs, assessing changes in recall values.
- Addressed potential sources of false positive and false negative alerts, such as time drifts or attack signatures, by integrating resyncing and retraining mechanisms to maintain reliable detection.

1.2 Organization of Thesis

The remainder of this thesis is structured as follows:

- Chapter 2: **Background and Literature Review** provides an overview of industrial control systems, specifically SCADA architectures. It discusses relevant defense techniques, including Intrusion Detection Systems, Finite State Machines, and the concepts of Digital Twins and Shadow Replicas, while surveying related work to highlight gaps in existing literature.

- Chapter 3: **Methodology** details the proposed system architecture. It defines the threat model, including system trust assumptions and adversarial capabilities, and presents the design of the tandem IIDS solution, which integrates NIDSs with a shadow replica.

- Chapter 4: **Evaluation** describes the experimental setup, including the benign and attack data collection procedures using the BEMOSS platform. It presents a preliminary scenario-based evaluation of locally executed attacks and a comprehensive evaluation using public IoT datasets to benchmark detection performance. Additionally, it discusses the findings and outlines potential directions for future work.

- Chapter 5: **Conclusions** summarizes the research contributions and implications of this study.

Chapter 2

Background and Literature Review

2.1 Industrial Control Systems and SCADA

Industrial control systems (ICS) encompass various interconnected technologies, and among these are SCADA systems [12]. SCADA is widely deployed for the supervisory management of remote assets, and the typical architecture of a modern SCADA system, as described in [10, 16], consists of three layers: the supervisory control, automatic control, and physical layers. The supervisory control layer (or control center) is responsible for monitoring the operation of SCADA systems by gathering data from field devices, performing control and supervisory tasks, and sending control commands to field controllers through the communication network. The automatic control layer (or regulatory control layer) is in charge of regulating the operation of physical processes based on control commands from the control center and sensor measurements from field devices. The physical processes are equipped with actuators (e.g., motors), sensors (e.g., temperature sensors), and protection devices (e.g., protective relays). The physical elements are controlled and monitored by the control center through the automatic control layer and the communication network.

[8, 11, 44] provide a detailed review of SCADA, its applications, and vulnerabilities. SCADA systems are prime targets for malicious actors due to their integration with IT networks [7]. Common attack vectors include malware, ransomware, or unauthorized system access via remote protocols. These attacks can enable command injection, alter device operations, and

disrupt physical processes. Vulnerabilities such as unencrypted communication processes [39], weak access control, and buffer overflows make SCADA environments susceptible to attack campaigns.

2.2 Intrusion Detection System Approaches for SCADA

Intrusion detection systems for SCADA and IoT can broadly be classified into three categories: signature-based [27], learning-based [26, 31], and hybrid approaches [9]. Signature-based systems rely on pre-established attack patterns to detect malicious activity, offering accuracy in detecting known attacks but limited capability against novel attacks or variations of known attacks. Learning-based systems, primarily leveraging machine learning models such as classifiers and neural networks, identify deviations from established system behavior. However, they are sensitive to the quality of training data, prone to overfitting, and may require extensive tuning. Hybrid systems combine both detection approaches, aiming to improve detection coverage and adaptability, particularly in SCADA/IoT environments where attack patterns and operational behaviors can vary significantly.

2.3 Finite State Machines

An FSM is a mathematical model consisting of a finite set of states and transitions [6]. The system occupies exactly one state at any given moment and progresses through states based on external input. FSMs are commonly used to model systems with event-driven behavior. For example, in a simple plug load FSM, when triggered by an input, the FSM transitions between the states *on* and *off*. This tool is a compact way of representing a continuously running system without considering the factor of time. For a complete definition of FSMs,

the initial states need to be known [17]. In our SCADA system, we use FSMs for intrusion detection because they model both normal and abnormal system behavior; and alert us to an attack when a deviation from the expected pattern occurs. FSMs are simple solutions to implement and also allow for real-time detection.

State machines are designed as Moore or Mealy machines according to [57]. In a Moore machine, the FSM goes to a state, performs an action (or a set of actions), and waits in that state for another event. In a Mealy machine, however, if the input changes, the FSM may perform an action. This action could involve changing a state or a similar operation. In this work, we use Moore state machines.

2.4 Digital Twins and Shadow Replicas

Originally introduced in [21] as a conceptual model for product life-cycle management, a digital twin is a virtual environment created in the original SCADA system that mirrors the logic and replicates the network interfaces of the system; and is used for securing industrial control systems [15]. [41] also explains this concept as a replicated system modeling the different functions of a physical system. It is able to change states as the physical model does because it has dual-direction communication, unlike similar solutions; digital models and digital shadows, that communicate in one direction [68]. Digital twins are now extensively used for monitoring, simulation, visualization, and testing of manufacturing systems. In addition, a variety of other applications can be achieved via big data analytics, summarized in [54]. Although these solutions have been shown to be effective for designing a variety of security solutions, such as IDSs, detecting misconfigurations, and penetration testing, they suffer drawbacks, like inaccuracies caused by software imperfections and high memory overhead with large data-generating SCADA systems.

The use of a shadow replica in our work can be fully described as a machine that runs the exact core software of the SCADA system and mirrors it at the logic level. The replica is on a local wired network, connected directly to the original SCADA system, allowing them to be synchronized with logs and also making them impervious to wireless network attacks. The shadow replica is a hardware and software solution that has similarities to the digital twin software solution. However, the replica offers several advantages, including additional security from its passive operation, enhanced accuracy due to improved synchronization, and service continuity. Furthermore, it minimizes the impact on the SCADA system, as the only source of overhead is the transmission of logs.

2.5 Gaps in Existing Literature

SCADA Security: Authors in [19] describe a range of security threats for industrial control systems, including repudiation, loss of availability, integrity, confidentiality, and authentication. An adversary can target the hardware, software, or network availability of SCADA systems. Security measures such as industrial intrusion detection systems, digital twins, firewalls, and antivirus software are discussed. The IEC technical committee attempted to redefine SCADA protocols for improved security with IEC 62351 [25], introducing end-to-end encryption to prevent attacks such as replay, MITM, spoofing, and packet injection. However, this solution is not backward compatible. In this work, we employ the combination of NIDSs and a shadow replica to secure industrial systems.

IDS Solutions: Signature-based NIDSs, such as Suricata, have been deployed in SCADA environments for the detection of known attacks [61]. [34] also demonstrates a signature-based solution to identify IoT attacks using manufacturer usage description specifications. In our work, Suricata is paired with Zeek to combine signature matching with protocol-aware

analysis, integrated with a shadow replica for improved detection in HTTP SCADA/IoT traffic.

Learning or behavior-based detection solutions for SCADA and IIoT model normal system behavior and flag deviations as intrusions. Examples include [23, 67], which apply neural network-based solutions to industrial and general IoT systems with high reported performance. Likewise, [55] presents a learning-based framework for IIoT evaluated on public datasets, while [22] develops a lightweight machine learning NIDS for IIoT. More recently, [52] proposed a federated learning approach for detecting attacks on IoT devices. In our work, we avoid learning-based approaches to eliminate the training overhead.

Recent work has explored diverse hybrid detection systems tailored to the constraints of SCADA and IoT environments. Examples include [59], which integrates 4 IIDSs in a portable, efficient design, and [60], which introduces an abstraction layer between IIDSs and industrial communication protocols for cross-system deployment. Stateful approaches have also been explored, such as FSM-based detection state machines for IEC 60870-5-104 [63], time-dependent finite state automaton for resource-constrained IoT devices [51], and deterministic finite automation for attack detection in Zigbee and Z-Wave platforms [66].

Shadow Replicas: Adding shadow replicas to a SCADA system is an alternative security mechanism for industrial systems. Our definition of a shadow replica slightly contrasts with that of the authors in [14], as our proposed replica does not need to reside on the same physical machine that hosts the original SCADA system. This concept is also similar to the concept of digital shadows in [5]. The digital shadows described by the authors are digital replicas of the physical SCADA system. However, they differ from our solution because shadow replicas can be separate physical systems that still match the states of the original SCADA system in real-time. Our shadow replica uses FSMs to model the interactions between the SCADA system and the smart devices under its control. We exploit

a more secure replica of the SCADA system compared to existing digital twin-based software solutions [15, 18, 62], resulting from the use of the FSM instead of a web front-end and direct communications with the devices.

Chapter 3

Methodology

3.1 Threat Model

Throughout this work, we focus on attackers whose goals are to infiltrate, monitor, or disrupt SCADA system operation. Attackers may be positioned physically adjacent to the SCADA system, putting them within range of the wireless networks. We also assume that attackers can use sniffers to extract Internet of Things (IoT) device events and information. Hence, we consider the SCADA system, including the IoT devices, as untrustworthy, whereas the shadow replica is assumed to be trustworthy. We do not account for the compromise of the shadow replica itself. However, in the event that the shadow replica assumes control of the primary SCADA controller due to a compromise, adversaries could potentially compromise the system. In the following, we define the trust boundaries in the system architecture, describe the classes of attacks considered, and list the limitations of the adversary.

3.1.1 System Trust Assumptions

To complement the attacker model, we define the following trust boundaries in our system.

Trusted Components:

- Shadow Replica: It operates in passive mode and is assumed to be uncompromised,

with direct access to logs and network traffic.

- NIDS: The network intrusion detection systems are assumed to be secure and correctly configured, and adversaries are not able to modify their detection rules.
- FSM Models and SCADA Logs: The FSM modeling framework and the SCADA logs stored on the replica are trusted and assumed not to be altered by adversaries, reflecting the current states of the devices.
- Operator: The human operator interface is assumed to be secure and not under adversarial control.

Untrusted Components:

- Primary SCADA Controller & IoT devices: These components are exposed to cyber-attacks and are considered untrusted due to their susceptibility to compromise.
- Wireless Communication Networks: All wireless links (WiFI, Zigbee, Bluetooth) are considered untrusted and are vulnerable to sniffing, replay, and DoS attacks.

3.1.2 Adversarial Capabilities

The classes of attacks we focus on in this work are inspired by the Open Web Application Security Project (OWASP) specified vulnerabilities listed for IoT applications [2]. We focus on wireless attacks where the attacker's objective is to target the system's integrity by inducing different sensor information or disrupting normal operations. From the relevant attacks listed below, we physically performed Data Manipulation, IP Spoofing, Code Injection, and DoS (SYN Flooding and Deauthentication) attacks on our system. We also theoretically discuss the unattempted attacks. SCADA/IoT systems are susceptible to these attacks as

they are very common and easy to launch without requiring a high level of expertise, thus posing a significant threat to existing smart infrastructure. We later show how our solution can detect and mitigate these attacks.

With **Data Manipulation** attacks, the adversary breaches the system and is positioned inside the network, able to issue commands such as changing or reading the states of the devices. This is caused by the lack of secured HTTP communication with the SCADA controller in plug-and-play smart devices[32, 37]. This attack can fly under the radar if performed well since it does not leave a substantial trace in the network traffic.

The adversary is also assumed to be positioned within the network for the **SYN-Flooding** DoS attack. An attacker can then initiate multiple connections with the devices directly using their IP addresses and port numbers, abusing the TCP three-way handshake [30, 37].

In **Deauthentication**, a different DoS attack, the adversary can be positioned outside the SCADA network and is able to collect the AP and devices' MAC addresses by sniffing the SCADA network traffic. With the address information, they can then issue Deauthentication frames to the devices and/or the SCADA controller, thereby disconnecting them from the wireless network and preventing communication [33].

For **IP Spoofing**, the adversary would need to be inside the network and obtain the IP address of the SCADA controller and devices from a MITM position. They could then send commands over the network to disrupt communication or read and modify the states of the devices with the new source IP address [37]. The network packet's source IP address is modified instead of the packet's payload, as in Data Manipulation.

Port Scanning and **Data Sniffing** are both information-gathering attacks. The former scans for open ports or services on smart devices, while the latter monitors all traffic in the network [37, 56]. These attacks are typically partnered with a secondary, more active attack

to affect the smart system directly.

For **Replay** attacks, a MITM position inside the network would need to be occupied by the adversary. They would then capture packets from the controller to the devices, re-transmit them later, and in doing so, make unwanted changes to a device's states [37].

Lastly, **Code Injection** enables the adversary to send malicious code to the controller or devices and retrieve or edit information from the system or the stored database of the devices [20, 37]. This is accomplished when they are positioned inside the network or if the devices are facing the outside network.

While the attacker is assumed to have significant capabilities within or adjacent to the SCADA network, some limitations apply. For instance, the attacker cannot compromise the shadow replica directly, as it is assumed to be isolated from the wireless interfaces and not exposed to the application layer. Secondly, the attacker cannot modify the shadow replica's logging mechanism, which is resistant to tampering for the scope of this work.

3.2 System Overview

We adopt the BEMOSS [38] platform as our SCADA controller due to its open-source nature, broad support for IoT devices, and compatibility with various communication protocols. While our implementation is currently limited to WiFi-based IoT devices, the approach remains applicable to other protocols, such as Bluetooth and Zigbee, since the targeted attacks are protocol-agnostic.

BEMOSS offers typical SCADA functionality, including device supervision and control, data logging, and a user interface representing the Human Machine Interface (HMI). Unlike traditional SCADA systems, it is entirely software-based and does not require dedicated hardware

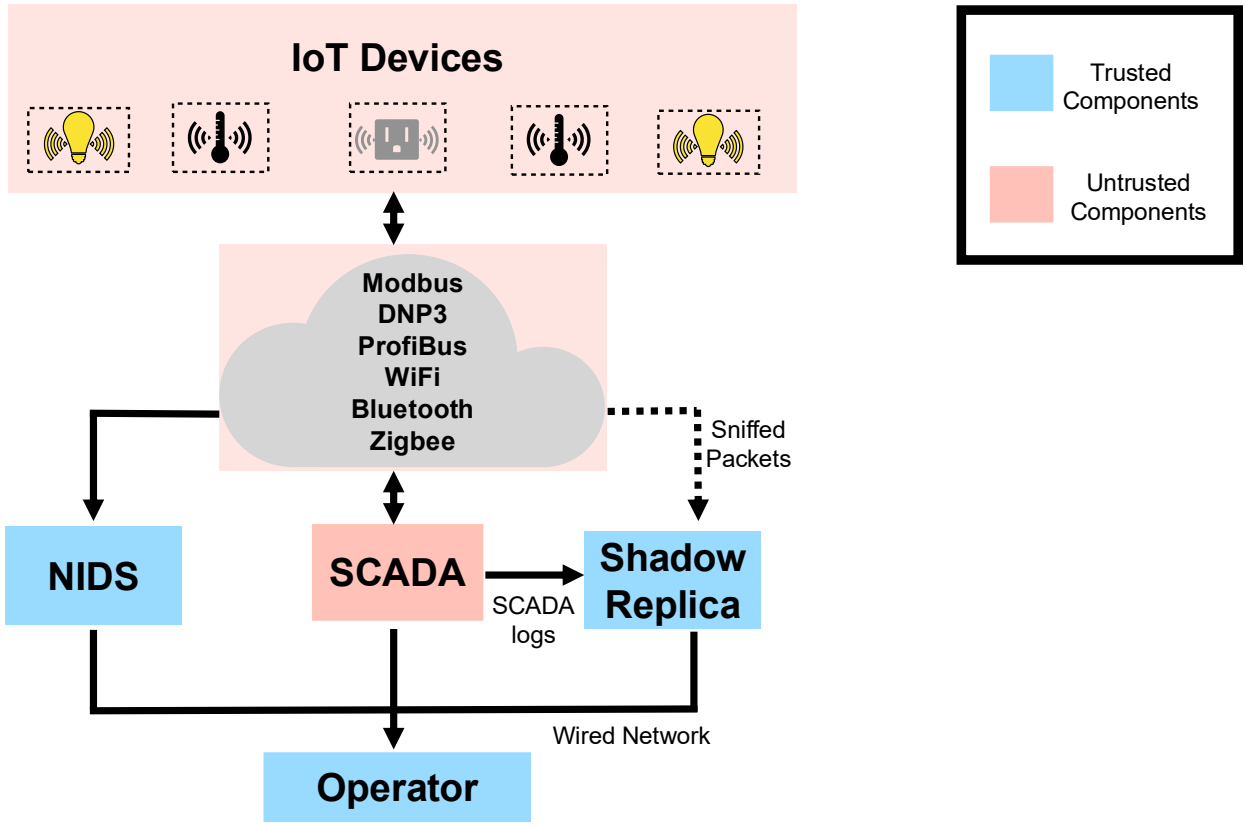

Figure 3.1: System architecture. Trusted components include the shadow replica, NIDS, and operator interface, while untrusted components include the IoT devices, SCADA controller, and wireless communication network.

components like Programmable Logic Controllers (PLCs) and Remote Transmission Units (RTUs). It supports a wide range of devices, such as sensors, HVAC systems, power meters, and plug or lighting controllers.

Our IIDS system architecture, illustrated in Figure 3.1, comprises two conventional NIDSs and a shadow replica based on FSMs. The framework is NIDS-agnostic and can operate with any standard NIDS. We select Suricata and Zeek as NIDSs for their distinct detection approaches: Suricata for its high-performance, multi-threaded threat detection, and Zeek for its event-driven traffic analysis. Employing two NIDSs enables cross-validation of alerts

and broader coverage against attacks, reducing blind spots inherent in relying on a single NIDS. Our design is limited to these two NIDSs to balance detection diversity with system complexity and processing overhead.

The components that comprise our system setup include:

- IoT Device I - 2 Belkin Wemo Plugs V3
- IoT Device II - 2 Radio Thermostat CT50 V1.94
- SCADA - BEMOSS V3.5 on Ubuntu 16.04 Virtual Machine with a Bridged and Host-only network interface.
- Suricata 6.0.10 and Zeek 4.0.6 NIDSs.
- Shadow Replica - BEMOSS V3.5 on Ubuntu 16.04 Virtual Machine with a Host-only network interface.
- Sentinel that monitors wireless traffic for the shadow replica - Raspberry Pi 4 running Kali Linux version 2021.2 with an external WiFi adapter.
- Attacker - Raspberry Pi 4 running Kali Linux version 2021.2 with an external WiFi adapter.
- Access Point (AP) - Netgear Nighthawk R8500 Router.

3.2.1 Attack Detection

The flow chart of an attack's detection can be seen in Figure 3.2. Our IIDS solution can detect attacks when they are launched and can also manage false alarms by resyncing the system. The system also allows for the NIDS to be retrained by using the attack's network

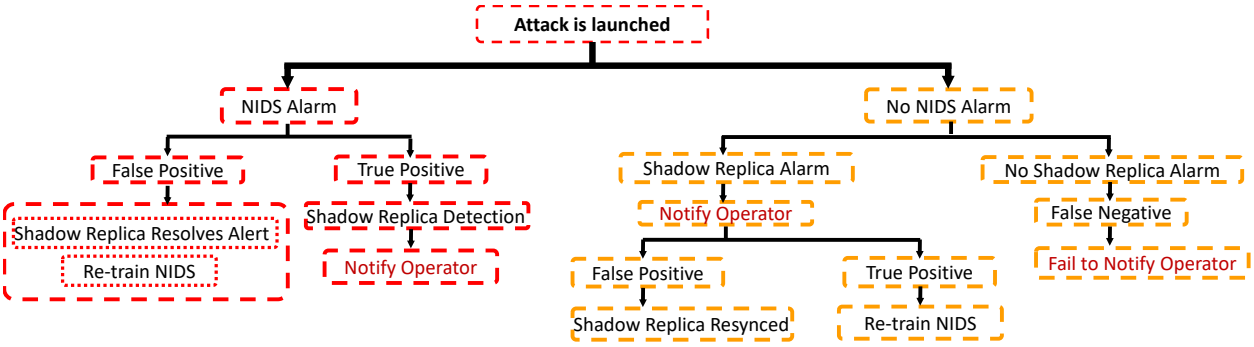

Figure 3.2: Attack detection flow diagram with tandem NIDS and shadow replica including the operator.

packets to update the signatures and rules of the NIDS to reduce possible errors. The operator (from Figure 3.1) is notified of events in the system as detection classifications occur.

Relying on the NIDS's detection alone does not provide a system with full security coverage, as there are attacks that can bypass it. That is why, in addition to the NIDS, to avoid the system being compromised, we implement a shadow replica that observes both the network packets and the SCADA logs. To detect an attack with the shadow replica's FSM, we compare the computed expected states with the logged actual states of the devices. If discrepancies are identified, the shadow replica raises an alert that points to an active attack or a reliability issue. The NIDSs operate simultaneously, and if there are alerts of an attack, those are also added to our attack profile. The flow chart from Figure 3.2 highlights the possible event scenarios using the information if an alert is raised by the shadow replica or NIDSs, and if there are no alerts at all.

In scenarios requiring takeover, the shadow replica maintains a continuously synchronized control state, enabling it to assume the role of the primary SCADA controller with minimal delay. Implemented as a virtual environment with an image of the primary BEMOSS controller and its preconfigured device connections, the replica is ready to interface with

the devices at any time. Its FSM continuously tracks the logical state of all connected devices, ensuring that upon activation, it can issue control commands without state mismatches. When an attack or reliability failure is suspected, the operator can switch to the shadow replica by activating its control mode, after which the replica assumes active control of the system. This capability extends beyond mere detection, providing service continuity by bypassing the compromised controller and acting as a backup SCADA system, even when the root cause, whether a security attack or reliability failure, cannot be immediately determined.

3.2.2 Attack Mitigation

Unlike traditional NIDSs that focus solely on detection, our IIDS solution also enables mitigation. When malicious activity is detected, the operator can isolate affected devices from the network, block or filter traffic based on the NIDS alert logs, and restore devices to their last known configurations using the state information from the replica's FSM. In cases such as DoS attacks on the controller or edge devices, the operator can observe the attack, remove the compromised components from the SCADA system, and preserve service continuity through either the primary controller or the replica.

3.2.3 State Tracking and FSM Modeling

To capture the application-level system state and complement the network system view, we use a deterministic approach with FSMs. To that end, we extract control commands for devices from the network data and associate them with the application-level state maintained by a shadow replica of the SCADA controller. This enables us to model both the initial state of the system and subsequent states by observing network communications that dictate

transitions of known and predefined FSM. The FSM is extracted from the application logic of the SCADA system. As network events occur, we track the estimated states of the devices as the output of the FSM.

Each network event prompts a comparison between the estimated states and the ground truth of the device states, which are recorded in the BEMOSS logs and then stored in the shadow replica. The BEMOSS logs consist of key information that includes an ID of the smart devices, any status changes or modifications to the device settings (such as turning a plug on or off and adjustments to HVAC settings), the user initiating the status check, and the timestamp of the event. These logs, originally housed in an Apache Cassandra database, are transferred to the shadow replica system through a direct-wired network connection in the virtual machine. The shadow replica also contains the network packet capture of the SCADA system, allowing for observation of variances between the states from the logs and the states seen in the network traffic. In a benign environment with no reliability issues, we expect to see no inconsistencies, as the system is a closed-loop. However, in the case of a variance, we can conclude that unwanted activity is occurring, as our system does not depend on error-prone probabilistic inference or behavior models. We ensure this by identifying the packet payload of the wireless control commands sent from the controller to each device before running experiments and hard-coding the payload into our FSM to identify changes in the states of each device.

Similarly, we cannot assume that the same network event has the same causal effect throughout the duration of the system. We establish that correlating events over time can lead to inconsistencies, as preliminary analysis depicted that the network estimations were late in comparison with the BEMOSS status updates for the first Plug. On the contrary, for the second Plug, the network estimations appeared early. Some of these effects are network latency and device manufacturer artifacts that prevent a pure time-based FSM from operating

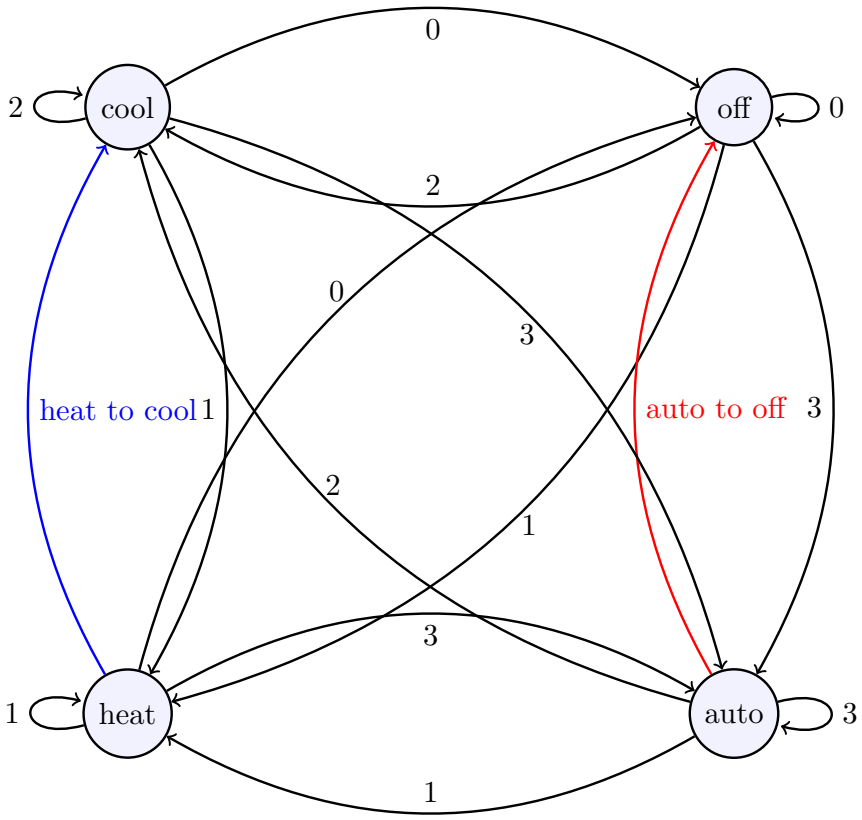

Figure 3.3: FSM model for heater in the Radio Thermostat representing a change in state from *heat* to *cool* (blue), and *auto* to *off* (red). These transitions happen as a result of the FSM identifying a HTTP payload in the network packets containing *tmode*:1 followed by *tmode*:2 (blue), and *tmode*:3 then *tmode*:0 (red).

properly. In our approach, we take this into consideration when we track system events in real-time, allowing for small time variations to exist by using time thresholds for network activity and application-level events. This means that we allow for inconsequential delays in system updates and responses that depend on the devices on the SCADA network, as not all devices have the same reaction times (within a small margin of time error of 2 seconds). This time error margin is small when compared to the time-scale of an attack and does not affect our attack detection and mitigation capabilities, but it does allow for error-free tracking of updates in the SCADA system consistently.

Figure 3.3 represents the generated FSM model for the Radio Thermostat heater, displaying all possible states of operation. As shown in the figure for the thermostat's heater, 0,1,2, and 3 represent *off*, *heat*, *cool*, and *auto* states, respectively. Similarly, for the thermostat's fan, 0,1, and 2 represent *off*, *on*, and *auto* states, respectively.

Chapter 4

Evaluation

4.1 Data Collection

This section describes the manual collection of both benign and attack data, generated through controlled experiments conducted using our SCADA and IoT testbed.

4.1.1 Benign Data Collection

BEMOSS is employed as the SCADA controller for two sets of devices: the Radio Thermostat CT50 V1.94 and Belkin Wemo Plugs V3. While installing the BEMOSS software, the device monitor was set to 1 second. The device monitor periodically queries the devices for their current states, allowing the system to record state changes, e.g., from *heat* to *cool* for a thermostat heater or *on* to *off* for plugs. Next, we perform testing to observe the changes in states stored in the local database. We also record the network traffic to and from the controller and the devices. In the benign trials, we change the states of all devices every 30 seconds for a testing duration of 1 hour. A chronological example of a trial looks like this:

- 12:00:00 - Plug 1 turned *on*, 12:00:30 - Thermostat 1's mode changes to *heat* 65°F

The shadow replica is in the local network, receiving the SCADA system's logs and the captured traffic. For the BEMOSS controller, a database stores state changes of all connected

devices. It also stores the user commands from the BEMOSS UI to control the devices. This is sent to the replica. After the hour-long trials, the database and the decrypted capture file logs are fed as inputs to the FSM to create the deterministic detection model.

4.1.2 Attack Data Collection

Attack trials are similar to the benign data collection experiments. However, attacks were executed during the normal BEMOSS operations.

For Data Manipulation, the pywemo [46] and radiotherm [42] python packages are used for their respective devices. We use the Attacker Raspberry Pi and send commands with known IP addresses to both thermostats to change their heater's thermostat modes to *cool* and the plugs to change their state to *on*.

For IP spoofing, the arpspoof [50] tool is used with the knowledge of all IP addresses in the SCADA system's network. We then spoof the IP addresses of the IoT devices and intercept the commands sent by the SCADA controller.

For SYN-Flooding, the IP addresses and port numbers for the devices are used to send SYN packets to the devices. After observing the original data collection capture files, we used port 80 for the thermostats and port 49153 for the plugs. We exploit the Attacker's hping3 [45] tool and flood the devices with 10,000 packets, each having a data size of 1,200 bytes.

For Deauthentication, the Attacker Raspberry Pi scans the network using airodump-ng from the aircrack-ng tools [1] and obtains the MAC addresses of the AP and the connected devices. We then continuously send deauthentication frames for 1 minute to each device using another tool, aireplay-ng.

The Attacker is positioned within the same network as the SCADA system for Code Injection.

Table 4.1: Attack detection and mitigation capabilities. Note that SR represents the shadow replica.

Attack	NIDS Only		NIDS + SR	
	Detect	**Mitigate**	**Detect**	**Mitigate**
Data Manipulation	x	x	✓	✓
SYN-Flooding	✓	x	✓	✓
Deauthentication	✓	x	✓	✓
IP Spoofing	x	x	✓	✓
Port Scanning	✓	x	✓	✓
Data Sniffing	x	x	x	x
Replay	x	x	✓	✓
Code Injection	x	x	✓	✓

We obtain the SCADA devices' IP addresses and open ports using general port scanning techniques. We then attempt a command injection exploit using Telnet and the Shellshock vulnerability [13].

These trials provided both the encrypted and decrypted packet capture files for each attack, as well as the states of each device from the SCADA database stored in the shadow replica. This data is then fed to the FSM, and the results help the running NIDS update its rules. The rules are updated by manually analyzing the capture files in order to find attack-relevant keywords or patterns to create new NIDS rules.

4.2 Results

In this section, we evaluate our integrated IIDS solution across multiple attack scenarios operating at different layers that influence data integrity and network connectivity. We focus on the following attacks: Data Manipulation, IP Spoofing, and Code Injection as threats to data integrity, and DoS attacks as those that disrupt device communication.

These attack classes are of particular interest because data integrity-related attacks lack a significant network footprint and bypass application-layer defenses, making them difficult to detect using traditional NIDSs. In contrast, connection-disruptive attacks tend to leave larger traces and are more easily detected by existing NIDS. We analyze the detection capabilities of our system, as listed in Table 4.1. We later highlight detection results on public IoT network datasets by comparing the confusion matrix results of Zeek and Suricata NIDSs with and without the proposed shadow replica, as seen in Table 4.2.

4.2.1 Preliminary Scenario-Based Evaluation

Shadow replicas replicate the SCADA controller using observable network traffic and device state logs. The network traffic is used solely for constructing the FSM to determine the states of the devices; thus, traffic alone cannot be used to detect all network-level attacks. However, when combined with NIDS alerts, the shadow replica can detect if a network or application attack has impacted the SCADA system's states, as observed in the attack below:

Data Manipulation: Note a sample of the performed attack from Figure 4.1.

- 4:55:33, 5:01:20, 5:16:10 - Plug 1 is turned *on*
- 4:52:31, 5:05:53, 5:13:27 - Plug 2 is turned *on*

The state estimations from network traffic are performed using deterministic FSMs. The goal is to perform intrusion detection by finding discrepancies between the expected (using the network traffic) and true states of the plug loads. For Plug 1 in this sample, attempts 1 and 3 cause the state of the device to be registered as *on* in the SCADA system. However, according to the network traffic, the state of the device should be *off*. The attempts for Data Manipulation are detected successfully. Attempt 2 is not causing a discrepancy because it

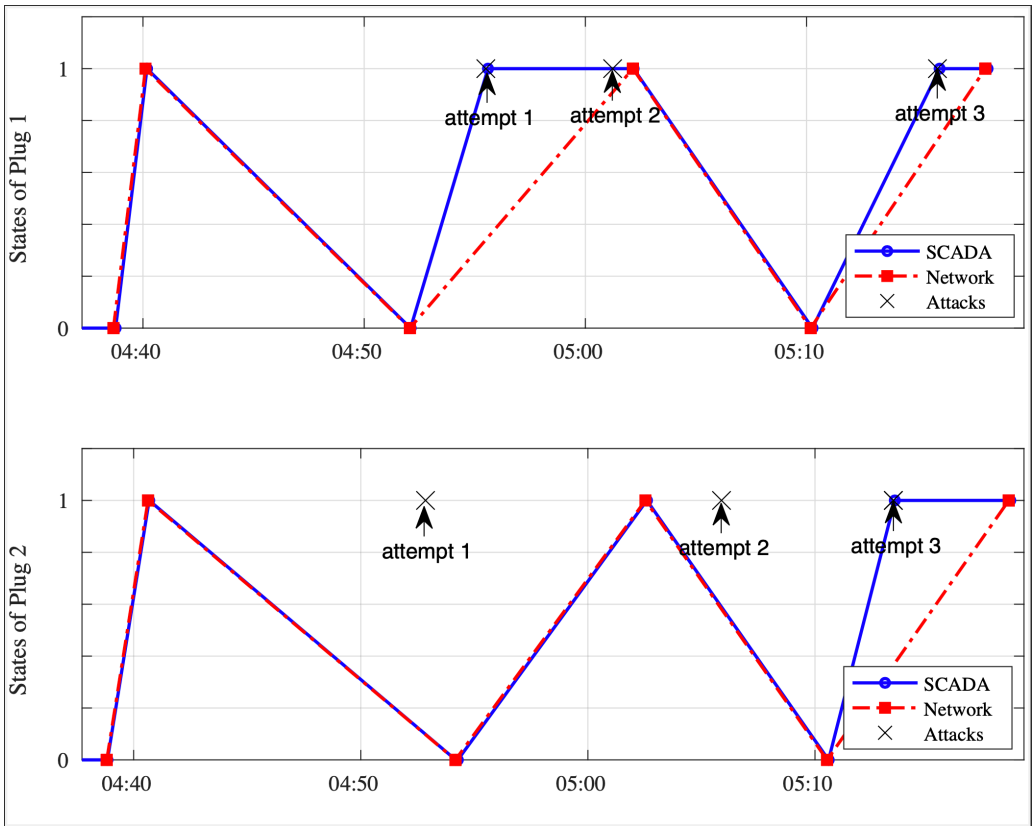

Figure 4.1: Impact of the Data Manipulation attack on the true states of the Plugs and their estimated states using the network traffic. Plug load states 0 and 1 represent *off* and *on* states respectively.

is, in essence, a wasted attempt in which the attacker tries to turn *on* a plug that is already *on*. The experiment follows, as attempts 1 and 2 on Plug 2 do not register in the SCADA system since they are also wasted attacks. Attempt 3 on Plug 2 is a successful attack that changes the state of the device from *off* to *on*. This attack, however, is detected by our solution.

Next, we launch Data Manipulation attacks on the Radio Thermostat's heater in the following order in Figure 4.2:

- 4:49:40, 4:58:30, 5:08:45 - Thermostat 1's mode is set to *cool*
- 5:03:25, 5:11:20 - Thermostat 2's mode is set to *cool*

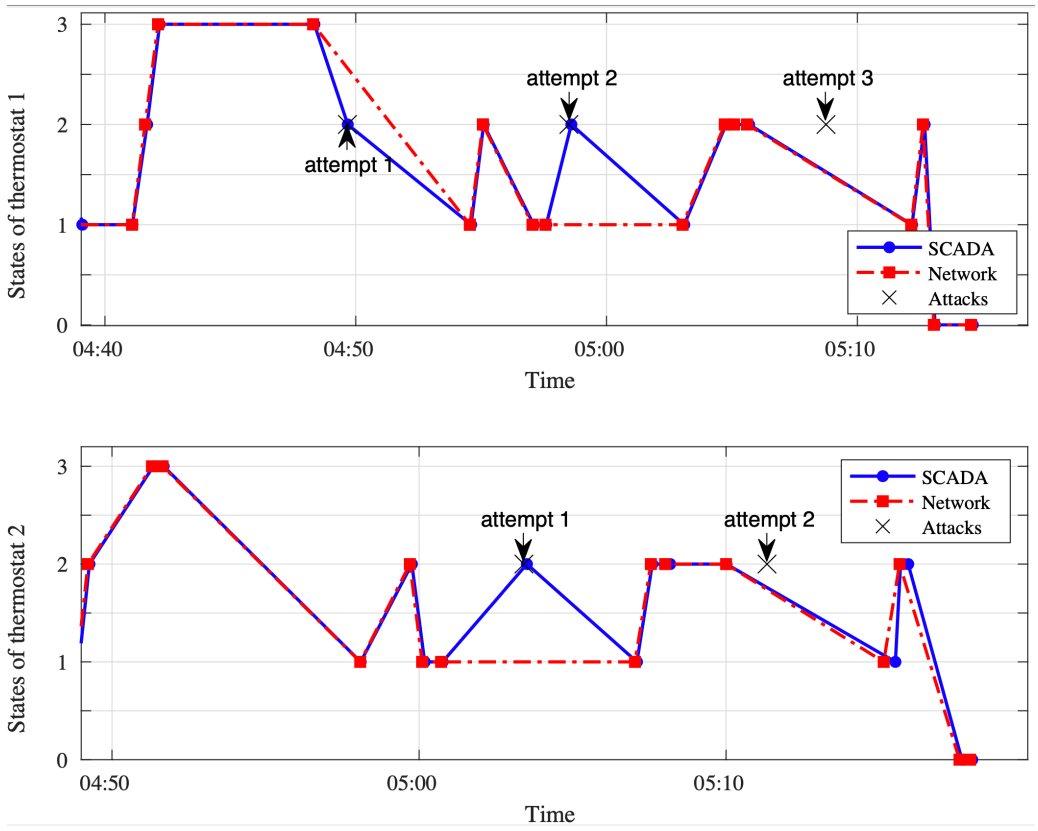

Figure 4.2: Impact of the Data Manipulation attack on the true states of the Radio Thermostats' heater and their estimated states using the network traffic. Thermostat states 0,1,2, and 3 represent *off*, *heat*, *cool*, and *auto* states respectively.

Radio Thermostat 1's heater demonstrates how the attacker's attempts 1 and 2 cause deviations in the system. At 4:49:40, the estimated state of the thermostat should be *auto*, while the SCADA system registers *cool*. At 4:58:30, the device should be at state *heat*, but its true state on the SCADA is *cool*. Both attempts lead to successfully executed attacks, effectively indicating that the deterministic nature of the data in the network traffic of SCADA systems allows our IIDS to identify both attacks successfully. Attempt 3 at 5:08:45, on the other hand, is another wasted trial by the attacker. For Radio Thermostat 2's heater, the attacker's first attempt at 5:03:25 successfully changes the state of the device from *heat* to *cool*. This occurs while our sentinel expects *heat* as the device's state. This disparity is used to detect the attack.

The following two classes of executed attacks from Table 4.1 also resulted in detection. **DoS**: Having executed these attacks, the network IDS alone was able to detect and report both the Deauthentication and SYN-Flooding attacks. For SYN-Flooding, the Suricata NIDS registered alerts noting the three-way handshake's incorrect seq and incorrect ack, while the Zeek IDS registered alerts with a content gap note for the TCP protocol. Similarly, in the case of Deauthentication, Suricata registered alerts noting errors with the three-way handshake, and Zeek registered alerts with content gaps. As is the nature of these attacks, the errors with the three-way handshake are a result of the execution of these DoS variants, which allows them to be detected by the NIDSs. **IP Spoofing**: Our approach detected a malicious attack on the system by using the logs and captured network traffic to identify discrepancies. The spoofing attack was blocking the SCADA controller from performing normal operations by communicating with the devices. The logs of the SCADA controller did not register any changes in the states of the devices because there was no network communication between them during the attack. The threat was detected by the NIDS and shadow replica solution due to the delay in normal operations of changing the device states.

The remaining classes of attacks were given a more comprehensive evaluation using robust, publicly available datasets in Section 4.2.2. **Code Injection**: In our attempt to execute this code injection attack variant, the attack script failed to launch on our deployed smart devices. This was neither captured in the network capture nor the SCADA logs. Similar to the IP Spoofing attack, our solution can detect a successful execution of the attack because the shadow replica is able to see the network traffic and the states of the devices but cannot run the code injection commands. As the SCADA controller is infected, it causes changes to the device states, and the disparities between the controller and the shadow replica alert the system to an attack. **Port Scanning**: This attack was not physically executed on our system, but similar to DoS attacks, there are common scanning signatures that allow NIDSs

Table 4.2: Tandem NIDS and Shadow Replica Detection on Datasets

Dataset	Attack Class	Attack Flows from GT	Zeek and Suricata				Shadow Replica
			TP	FP	FN	Recall	Recall
Bot-IoT	DDoS	112	23	9	89	20.54%	-
	DoS	109	24	6	85	22.02%	-
	Scanning	128	48	37	80	37.50%	-
	Theft	10	5	9	5	50.00%	100%
TON_IoT	DDoS	131	47	88	84	35.88%	-
	DoS	169	74	101	95	43.79%	-
	Scanning	1545	66	213	1479	4.27%	-
	Injection	118	59	63	59	50.00%	100%
	MITM	178	9	64	169	5.06%	100%
	Backdoor	20	4	74	16	20.00%	100%
	XSS	120	44	58	76	36.67%	100%
	Ransomware	10	2	33	8	20.00%	100%
	Password	136	39	76	97	28.68%	100%

to detect it. When launched, it would be identifiable by our tandem solution once targeting victim devices. **Data Sniffing**: While not physically executed, it is a passive attack that does not leave traces on either the network or the logical flow of events in the system; hence, our tandem IIDS would fail to detect it. **Replay**: This attack can be detected by the shadow replica receiving all the network data and states of devices. A replayed packet to change states will be identified as there is a timestamp from the controller's outgoing network packet to a device, and there will be an identifiable time delay when the device changes its states. The FSM can expose either an attack or device failure in this case.

4.2.2 Comprehensive Dataset Evaluation

We further assess the effectiveness of the proposed NIDS and shadow replica solution by analyzing the confusion matrix results for various attack types using two public IoT datasets: BoT-IoT [28] and TON_IoT [35]. Bot-IoT comprises both simulated and legitimate IoT network traffic that covers DoS, Distributed DoS (DDoS), Scanning, and Theft attack categories. TON_IoT includes telemetry data from IoT/IIoT sensors sharing similar attack

categories with Bot-IoT, as well as Backdoor, Injection, Cross site scripting (XSS), Ransomware, and Password attacks.

For this evaluation, 359 attack flows were extracted from Bot-IoT and 2,427 from TON_IoT. Ground truth labels were provided within the datasets based on annotated capture files. Network packets associated with each attack were then aggregated into flows for further analysis.

Zeek and Suricata NIDSs were each executed on the attack capture files. Suricata was configured using rules from multiple open-source sources, including Emerging Threats Open, OISF Traffic ID, Etnetera Aggressive IP Blacklist, SSL Blacklist, JA3 Fingerprints, Lateral, Win-Malware, and Hunting. Zeek was deployed with its `test-all-policy` script. Alerts were extracted from the `alert-debug` for Suricata and the `notice` logs for Zeek.

To evaluate performance, alerts were matched with the ground truth by converting attack packets into bi-directional flow tuples containing the source and destination IP addresses and protocols. *True positives* (TPs) were counted as flows from the NIDS's alerts that were classified as attacks in the ground truth. *False positives* (FPs) were counted as NIDS alerts not classified as attacks in the ground truth. *False negatives* (FNs) were counted as flows classified as attacks in the ground truth but not detected by the NIDSs.

The results are shown in Table 4.2. Recall is used as the primary evaluation metric, representing the proportion of correctly detected attack flows out of all labeled attack flows. Notably, the designed shadow replica framework demonstrates potential in enhancing recall, particularly with application-layer attacks. Based on the modeled behavior and framework configuration, the estimated mean recall increases by approximately 233% when using the combined NIDSs and shadow replica system compared to using the NIDSs alone.

In the Bot-IoT dataset, combined Zeek and Suricata NIDSs achieved recall rates of 21%,

22%, and 38% for DDoS, DoS, and Scanning attacks, respectively. These attacks were solely detected by the NIDSs. However, Theft attacks have a recall score of 50% with the NIDSs alone and are projected to show a 100% improvement with the implementation of the shadow replica due to the replica's detection capabilities for these application-layer attacks.

In the TON_IoT dataset, the NIDSs detected the network-layer attacks (DoS, DDoS, and Scanning) with recall scores of 44%, 36%, and 4%, respectively. For the remaining six application-layer attack classes (Injection, MITM, Backdoor, XSS, Ransomware, Password), the shadow replica is estimated to improve detection. Across these categories, the modeled mean recall improves by approximately 274%.

4.3 Discussion

4.3.1 Threats to Validity

In evaluating our integrated intrusion detection and mitigation framework, which combines both network intrusion detection systems and shadow replicas, it is important to acknowledge several factors that may affect the validity of our results.

First, the relatively low recall values of Zeek and Suricata NIDSs represent a threat to validity, as their detection capabilities are inherently constrained by the quality and specificity of the attack detection signature scripts employed. Nevertheless, the primary focus of this work is to demonstrate the performance enhancement achieved by augmenting existing NIDSs with a shadow replica.

Second, the current evaluation was conducted using a selected subset of IoT devices, which may not fully capture the diversity and operational complexity of broader industrial control environments. As a result, the applicability of our findings to scenarios involving pro-

grammable logic controllers, remote terminal units, and other operational technologies may be limited.

Finally, the manual construction of finite state machines for individual IoT devices introduces a scalability bottleneck. While this approach is viable for the small-scale setup involving two device categories used in this study, it does not scale well to larger and heterogeneous environments.

4.3.2 Future Work

Several promising directions remain to be explored in order to extend the applicability and resilience of the proposed framework.

To address the scalability bottleneck introduced by manual FSM design, future work will explore automated FSM inference techniques such as those proposed in [48]. Such automation would enable broader adoption across diverse device categories with minimal manual effort.

An ongoing challenge is to ensure that the shadow replica remains isolated from corruption by adversaries. While its lack of network connectivity reduces exposure to remote attacks, additional protection can be achieved by limiting the replica's software stack strictly to FSM execution. Determining the appropriate scope of this stack may be left to the operator, based on deployment needs.

The proposed IIDS solution also remains susceptible to firmware-level attacks that could compromise both the SCADA controller and its replicas. A possible mitigation is to diversify firmware or SCADA implementations across replicas. However, this may introduce new challenges related to synchronization inconsistencies, potentially reducing reliability. As a future direction, we plan to integrate SCADA controllers with strict access control mechanisms for connected devices to limit the risk of such adversarial compromise.

Chapter 5

Conclusions

This thesis proposes a tandem IIDS approach that combines two traditional network intrusion detection systems with a shadow replica. The replica, augmented with FSMs, processes the network events and device state of the primary SCADA system. Unlike conventional NIDSs or digital twins, our design enables the detection of attacks across both network and application layers. In the event of a successful compromise, the replica can assume control, maintaining operations through the synchronized device states with the primary SCADA system. We implemented a prototype using the open-source SCADA platform BEMOSS to manage and control smart devices, and evaluated it under both benign and malicious conditions through locally executed attacks relevant to SCADA and IoT systems. Additional tests on public IoT datasets provided a baseline for comparison. Results show that incorporating the shadow replica alongside Zeek and Suricata can improve recall values in our evaluation. Finally, the defense mechanism supports retraining and resyncing to mitigate model drift, helping to maintain detection performance over time.

Bibliography

[1] aircrack-ng(1) - Linux man page. https://linux.die.net/man/1/aircrack-ng.

[2] Owasp internet of things | owasp foundation. https://owasp.org/www-project-internet-of-things/.

[3] Desigo CC. https://www.siemens.com/global/en/products/buildings/automation/desigo/building-management/desigo-cc.html.

[4] Metasys Building Automation System. https://www.johnsoncontrols.com/building-automation-and-controls/building-management/building-automation-systems-bas.

[5] Ayman AboElHassan, Ahmed Sakr, and Soumaya Yacout. A framework for digital twin deployment in production systems. In *Advances in Automotive Production Technology–Theory and Application*, pages 145–152. Springer, 2021.

[6] Arash Ardakani, Amir Ardakani, and Warren Gross. Training linear finite-state machines. *Advances in Neural Information Processing Systems*, 33:7173–7183, 2020.

[7] Muhammad Muzamil Aslam, Ali Tufail, Rosyzie Anna Awg Haji Mohd Apong, Liyanage Chandratilak De Silva, and Muhammad Taqi Raza. Scrutinizing security in industrial control systems: An architectural vulnerabilities and communication network perspective. *IEEE Access*, 12:67537–67573, 2024.

[8] Tom Bartman and Kevin Carson. Securing communications for scada and critical industrial systems. In *2016 69th Annual Conference for Protective Relay Engineers (CPRE)*, pages 1–10. IEEE, 2016.

[9] Aitor Belenguer, Javier Navaridas, and Jose A. Pascual. A review of federated learning in intrusion detection systems for iot, 2022.

[10] Stuart A Boyer. *SCADA: supervisory control and data acquisition*. International Society of Automation, 2009.

[11] Rodrigo Chandia, Jesus Gonzalez, Tim Kilpatrick, Mauricio Papa, and Sujeet Shenoi. Security strategies for scada networks. In *International Conference on Critical Infrastructure Protection*, pages 117–131. Springer, 2007.

[12] Mauro Conti, Denis Donadel, and Federico Turrin. A survey on industrial control system testbeds and datasets for security research. *IEEE Communications Surveys & Tutorials*, 23(4):2248–2294, 2021.

[13] Digi Ninja. Shellshock and the Telnet USER Variable. https://digi.ninja/blog/telnet_shellshock.php.

[14] Tobias Distler. Byzantine fault-tolerant state-machine replication from a systems perspective. *ACM Comput. Surv.*, 54(1), feb 2021.

[15] Matthias Eckhart and Andreas Ekelhart. A specification-based state replication approach for digital twins. In *Proceedings of the 2018 Workshop on Cyber-Physical Systems Security and Privacy*, pages 36–47, 2018.

[16] Lionel Fillatre, Igor Nikiforov, Peter Willett, et al. Security of scada systems against cyber–physical attacks. *IEEE Aerospace and Electronic Systems Magazine*, 32(5):28–45, 2017.

[17] Robert M Fuhrer and Steven M Nowick. *Sequential optimization of asynchronous and synchronous finite-state machines: Algorithms and tools*. Springer Science & Business Media, 2012.

[18] Christian Gehrmann and Martin Gunnarsson. A digital twin based industrial automation and control system security architecture. *IEEE Transactions on Industrial Informatics*, 16(1):669–680, 2019.

[19] Sagarika Ghosh and Srinivas Sampalli. A survey of security in scada networks: Current issues and future challenges. *IEEE Access*, 7:135812–135831, 2019.

[20] Sagarika Ghosh and Srinivas Sampalli. A survey of security in scada networks: Current issues and future challenges. *IEEE Access*, 7:135812–135831, 2019.

[21] Michael Grieves. Digital twin: manufacturing excellence through virtual factory replication. *White paper*, 1:1–7, 2014.

[22] Eric Gyamfi and Anca Delia Jurcut. Novel online network intrusion detection system for industrial iot based on oi-svdd and as-elm. *IEEE Internet of Things Journal*, 10(5): 3827–3839, 2023.

[23] Zakaria Abou El Houda, Bouziane Brik, and Lyes Khoukhi. "why should i trust your ids?": An explainable deep learning framework for intrusion detection systems in internet of things networks. *IEEE Open Journal of the Communications Society*, 3: 1164–1176, 2022.

[24] The White House. FACT SHEET: Biden-Harris Administration Announces National Cybersecurity Strategy. URL https://www.whitehouse.gov/briefing-room/statements-releases/2023/03/02/fact-sheet-biden-harris-administration-announces-national-cybersecurity-strategy.

[25] IEC. Power systems management and associated information exchange - data and communications security. Standard, TC 57, March 2020.

[26] Kevin I Jones, Helge Janicke, Christian Facchi, and Leandros A Maglaras. Introduction to the special issue of the journal of information security and applications on" ics & scada cyber security". *J. Inf. Secur. Appl.*, 34:152, 2017.

[27] Mohamad Kaouk, Jean-Marie Flaus, Marie-Laure Potet, and Roland Groz. A review of intrusion detection systems for industrial control systems. In *2019 6th International Conference on Control, Decision and Information Technologies (CoDIT)*, pages 1699–1704, 2019. doi: 10.1109/CoDIT.2019.8820602.

[28] Nickolaos Koroniotis, Nour Moustafa, Elena Sitnikova, and Benjamin Turnbull. Towards the development of realistic botnet dataset in the internet of things for network forensic analytics: Bot-iot dataset, 2018.

[29] Moshe Kravchik and Asaf Shabtai. Detecting cyber attacks in industrial control systems using convolutional neural networks. In *Proceedings of the 2018 Workshop on Cyber-Physical Systems Security and PrivaCy*, page 72–83. Association for Computing Machinery, 2018. ISBN 9781450359924.

[30] Lulu Liang, Kai Zheng, Qiankun Sheng, and Xin Huang. A denial of service attack method for an iot system. In *2016 8th International Conference on Information Technology in Medicine and Education (ITME)*, pages 360–364, 2016.

[31] Chih-Yuan Lin and Simin Nadjm-Tehrani. Timing patterns and correlations in spontaneous {SCADA} traffic for anomaly detection. In *22nd International Symposium on Research in Attacks, Intrusions and Defenses ({RAID} 2019)*, pages 73–88, 2019.

[32] Haoyu Liu, Tom Spink, and Paul Patras. Uncovering security vulnerabilities in the belkin wemo home automation ecosystem. In *2019 IEEE International Conference on Pervasive Computing and Communications Workshops (PerCom Workshops)*, pages 894–899, 2019.

[33] Karim Lounis and Mohammad Zulkernine. Attacks and defenses in short-range wireless technologies for iot. *IEEE Access*, 8:88892–88932, 2020.

[34] Luca Morgese Zangrandi, Thijs Van Ede, Tim Booij, Savio Sciancalepore, Luca Allodi, and Andrea Continella. Stepping out of the mud: Contextual threat information for iot devices with manufacturer-provided behavior profiles. In *Proceedings of the 38th Annual Computer Security Applications Conference*, ACSAC '22, page 467–480, 2022.

[35] Nour Moustafa. A new distributed architecture for evaluating ai-based security systems at the edge: Network ton_iot datasets. *Sustainable Cities and Society*, 72:102994, 2021.

[36] Eugène David Ngangue Ndih and Soumaya Cherkaoui. On enhancing technology coexistence in the iot era: Zigbee and 802.11 case. *IEEE Access*, 2016.

[37] Abhijeet C Panchal, Vijay M Khadse, and Parikshit N Mahalle. Security issues in iiot: A comprehensive survey of attacks on iiot and its countermeasures. In *2018 IEEE Global Conference on Wireless Computing and Networking (GCWCN)*, pages 124–130. IEEE, 2018.

[38] M. Pipattanasomporn, M. Kuzlu, W. Khamphanchai, A. Saha, K. Rathinavel, and S. Rahman. Bemoss: An agent platform to facilitate grid-interactive building operation with iot devices. In *2015 IEEE Innovative Smart Grid Technologies - Asia (ISGT ASIA)*, 2015.

[39] Dimitrios Pliatsios, Panagiotis Sarigiannidis, Thomas Lagkas, and Antonios G Sarigiannidis. A survey on scada systems: secure protocols, incidents, threats and tactics. *IEEE Communications Surveys & Tutorials*, 22(3):1942–1976, 2020.

[40] Michalis Polychronakis and Kostas G. Anagnostakis. An empirical study of real-world

polymorphic code injection attacks. In *2nd USENIX Workshop on Large-Scale Exploits and Emergent Threats (LEET 09)*, Boston, MA, 2009.

[41] Cheng Qian, Xing Liu, Colin Ripley, Mian Qian, Fan Liang, and Wei Yu. Digital twin—cyber replica of physical things: Architecture, applications and future research directions. *Future Internet*, 14(2):64, 2022.

[42] radiothermostat. radiothermostat. https://www.radiothermostat.com/.

[43] Donald Ray and Jay Ligatti. Defining code-injection attacks. In *Proceedings of the 39th Annual ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages*, POPL '12, page 179–190, 2012.

[44] Sagar Samtani, Shuo Yu, Hongyi Zhu, Mark Patton, and Hsinchun Chen. Identifying scada vulnerabilities using passive and active vulnerability assessment techniques. In *2016 IEEE Conference on Intelligence and Security Informatics (ISI)*, pages 25–30. IEEE, 2016.

[45] Sanfilippo, Salvatore. hping3(8) - Linux man page. https://linux.die.net/man/8/hping3 ,.

[46] Severance, Eric. pywemo 0.6.7. https://pypi.org/project/pywemo/ ,.

[47] Shachar Menashe, Or Peles, Ori Hollander. Log4j Log4Shell 0-Day Vulnerability: All You Need To Know. https://jfrog.com/blog/log4shell-0-day-vulnerability-all-you-need-to-know/.

[48] Zhan Shu and Guanhua Yan. Iotinfer: Automated blackbox fuzz testing of iot network protocols guided by finite state machine inference. *IEEE Internet of Things Journal*, 9 (22):22737–22751, 2022.

[49] Yingbo Song, Michael E. Locasto, Angelos Stavrou, Angelos D. Keromytis, and Salvatore J. Stolfo. On the infeasibility of modeling polymorphic shellcode. In *Proceedings of the 2007 ACM Conference on Computer and Communications Security, CCS 2007, Alexandria, Virginia, USA, October 28-31, 2007*, pages 541–551, 2007.

[50] Song, Dug. arpspoof(8) - Linux man page. https://linux.die.net/man/8/arpspoof ,.

[51] Venkatraman Subbarayalu, B Surendiran, and P Arun Raj Kumar. Hybrid network intrusion detection system for smart environments based on internet of things. *The Computer Journal*, 62(12):1822–1839, 2019.

[52] Shihua Sun, Pragya Sharma, Kenechukwu Nwodo, Angelos Stavrou, and Haining Wang. Fedmade: Robust federated learning for intrusion detection in iot networks using a dynamic aggregation method. In *International Conference on Information Security*, pages 286–306. Springer, 2024.

[53] suricata. suricata. https://suricata.io/.

[54] Fei Tao, Weiran Liu, Jianhua Liu, X Liu, Q Liu, T Qu, T Hu, Z Zhang, et al. Digital twin and its potential application exploration. *Computer Integrated Manufacturing Systems*, 24(1):1–18, 2018.

[55] Akbar Telikani, Jun Shen, Jie Yang, and Peng Wang. Industrial iot intrusion detection via evolutionary cost-sensitive learning and fog computing. *IEEE Internet of Things Journal*, 9(22):23260–23271, 2022.

[56] Robert Udd, Mikael Asplund, Simin Nadjm-Tehrani, Mehrdad Kazemtabrizi, and Mathias Ekstedt. Exploiting bro for intrusion detection in a scada system. In *Proceedings of*

the 2nd ACM International Workshop on Cyber-Physical System Security, pages 44–51, 2016.

[57] Ferdinand Wagner. *Modeling software with finite state machines: a practical approach*. Auerbach Publications, 2006.

[58] Konrad Wolsing, Eric Wagner, and Martin Henze. *Facilitating Protocol-Independent Industrial Intrusion Detection Systems*, page 2105–2107. 2020.

[59] Konrad Wolsing, Lea Thiemt, Christian van Sloun, Eric Wagner, Klaus Wehrle, and Martin Henze. Can industrial intrusion detection be simple? In *Computer Security–ESORICS 2022: 27th European Symposium on Research in Computer Security, Copenhagen, Denmark, September 26–30, 2022, Proceedings, Part III*, pages 574–594. Springer, 2022.

[60] Konrad Wolsing, Eric Wagner, Antoine Saillard, and Martin Henze. Ipal: Breaking up silos of protocol-dependent and domain-specific industrial intrusion detection systems. In *Proceedings of the 25th International Symposium on Research in Attacks, Intrusions and Defenses*, RAID '22, page 510–525, 2022.

[61] Kevin Wong, Craig Dillabaugh, Nabil Seddigh, and Biswajit Nandy. Enhancing suricata intrusion detection system for cyber security in scada networks. In *2017 IEEE 30th Canadian Conference on Electrical and Computer Engineering (CCECE)*, pages 1–5, 2017.

[62] Qinghua Xu, Shaukat Ali, and Tao Yue. Digital twin-based anomaly detection in cyber-physical systems. In *2021 14th IEEE Conference on Software Testing, Verification and Validation (ICST)*, pages 205–216. IEEE, 2021.

[63] Y. Yang, K. McLaughlin, S. Sezer, Y.B. Yuan, and W. Huang. Stateful intrusion

detection for iec 60870-5-104 scada security. In *2014 IEEE PES General Meeting | Conference & Exposition*, pages 1–5, 2014.

[64] Junjie Yin, Zheng Yang, Hao Cao, Tongtong Liu, Zimu Zhou, and Chenshu Wu. A survey on bluetooth 5.0 and mesh: New milestones of iot. *ACM Trans. Sen. Netw.*, 15 (3), May 2019.

[65] zeek. zeek. https://zeek.org/.

[66] Wei Zhang, Yan Meng, Yugeng Liu, Xiaokuan Zhang, Yinqian Zhang, and Haojin Zhu. Homonit: Monitoring smart home apps from encrypted traffic. In *Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security*, pages 1074–1088, 2018.

[67] Yichi Zhang, Chunhua Yang, Keke Huang, and Yonggang Li. Intrusion detection of industrial internet-of-things based on reconstructed graph neural networks. *IEEE Transactions on Network Science and Engineering*, pages 1–12, 2022.

[68] Haifeng Zhou, Mohan Li, Yanbin Sun, Zhihong Tian, and Lei Yun. Digital twin based cyber range for industrial internet of things. *IEEE Consumer Electronics Magazine*, pages 1–11, 2022.