Attack Resilient Cloud-based Industrial Control Systems

Licentiate Thesis

Fatemeh Akbarian

Department of Electrical and
Information Technology
Lund, April 2022

Fatemeh Akbarian
Department of Electrical and Information Technology
Lund University
Ole Römers Väg 3, 223 63 Lund, Sweden

Series of Licentiate and Doctoral Theses
ISSN 1654-790X, No. 145
ISBN 978-91-8039-219-8 (printed)
ISBN 978-91-8039-220-4 (digital)

This thesis is typeset using $\LaTeX\,2_\varepsilon$.

Printed by Tryckeriet i E-huset, Lund University, Lund, Sweden, 2022.

Dedicated to my family

Abstract

Industrial control systems (ICSs) are a significant part of industry and they play an important role in monitoring and controlling industrial processes. Traditionally, ICSs have been isolated from the Internet, and thereby secured from various Internet-based security threats. In recent years, since the cloud can provide huge advantages regarding storage and computing resources, industry has been motivated to move industrial control systems to the cloud. However, when ICSs are moved to the cloud, they are inevitably exposed to increasing security threats, which can lead to severe degradation of the system performance or system failures. Moving control systems to the cloud can enable attackers to infiltrate the system and establish an attack that can lead to damages and disruptions with potentially catastrophic consequences. Therefore, some security measures are necessary to detect these attacks in a timely manner and mitigate the impact of them.

In the work presented in this thesis, we mainly explore the security challenges of cloud-based industrial control systems and we propose a security framework for these systems that can make them resilient against attacks. Our proposed framework includes attack detection methods that can detect attacks in a timely manner. Also, the framework includes mitigation methods that can mitigate the impact of the attack on the system when an attack has been detected. So, by using this framework, an industrial plant can be maintained operational with an acceptable performance during an attack.

Our solutions are validated on a real testbed, where the capabilities are evaluated by subjecting the system to a set of attacks.

Preface

This licentiate thesis concludes my work as a licentiate candidate and is comprised of two parts. The first part gives an overview of the research field in which I have been working during my licentiate and a brief summary of my contribution in it. The second part is composed of four included papers that constitute my main scientific work:

INCLUDED PAPERS

The following papers form the main body of this thesis and the respective published or draft versions are appended in the back.

Paper I: FATEMEH AKBARIAN, EMMA FITZGERALD, MARIA KIHL, "Intrusion Detection in Digital Twins for Industrial Control Systems", *The 28th International Conference on Software, Telecommunications and Computer Networks (SoftCOM)*, Hvar, Croatia, Sept. 2020.

Paper II: FATEMEH AKBARIAN, WILLIAM TÄRNEBERG, EMMA FITZGERALD, MARIA KIHL, "A Security Framework in Digital Twins for Cloud-based Industrial Control Systems: Intrusion Detection and Mitigation", *The 26th IEEE International Conference on Emerging Technologies and Factory Automation (ETFA)*, Västerås, Sweden, Sep. 2021.

Paper III: FATEMEH AKBARIAN, WILLIAM TÄRNEBERG, EMMA FITZGERALD, MARIA KIHL, "A cloud-control system equipped with intrusion detection and mitigation", *Conference on Networked Systems 2021 (NetSys 2021)*, Lübeck, Germany, Sep. 2021.

Paper IV: <u>Fatemeh Akbarian</u>, William Tärneberg, Emma Fitzgerald, Maria Kihl, "Attack resilient cloud control systems", *IEEE Open Journal of Industry Applications*, submitted, Mar. 2022.

OTHER CONTRIBUTIONS

During my licentiate, I have also contributed to the following publications, which are not included in the thesis:

Paper V: <u>Fatemeh Akbarian</u>, Emma Fitzgerald, Maria Kihl, "Synchronization in digital twins for industrial control systems", *16th Swedish National Computer Networking Workshop (SNCNW 2020)*, Kristianstad, Sweden, May. 2020.

Paper VI: <u>Fatemeh Akbarian</u>, Amin Ramezani, Mohammad-Taghi Hamidi-Beheshti, Vahid Haghighat, "Advanced algorithm to detect stealthy cyber attacks on automatic generation control in smart grid", *IET Cyber-Physical Systems: Theory Applications*, vol. 5, Iss. 4, pp. 351–358, Dec. 2020.

Paper VII: <u>Fatemeh Akbarian</u>, William Tärneberg, Emma Fitzgerald, Maria Kihl, "Detection and mitigation of deception attacks on cloud-based industrial control systems", *25th Conference on Innovation in Clouds, Internet and Networks (ICIN)*, Accepted, Paris, France, Mar. 2022.

Acknowledgments

Over the course of my PhD journey, I have been lucky to have people around that without their support and help this work could not have been completed. First and foremost, I would like to express my deepest gratitude to Maria Kihl, my main supervisor. Her ongoing guidance, encouragement, trust and valuable critiques gave me confidence to successfully overcome challenges in my PhD studies.

I am deeply grateful to my co-supervisor, William Tärneberg for sharing his knowledge and rigorous attitude in research that has shaped my scientific approach in solving my research questions. I would like to show my greatest appreciation to Emma Fitzgerald, also my co-supervisor. Her talent and also her enthusiasm to discuss scientific problems and sincere support has always been inspiring to me.

I would like to thank all the collaborators and the ones that have helped me in my research. A big thanks to all my friends and colleagues in Networked systems lab. You have all helped to create an excellent research environment. I have enjoyed all your company and appreciate all your supports during my difficult times. Also, I want to thank dear Zahra and Haorui. Their help from the first day of my arrival created an unforgettable memory.

My heartfelt appreciation goes to my family who always believed in me. My parents, my brothers, and my sister, without your endless support and kindness it would have been impossible to be where I am today.

Finally and most importantly, I would like to give my special thanks to my dear husband, Vahid. His unconditional love and endless support during both difficult and happy times has given me the passion and strength to bring my PhD journey to this point.

Contents

Acronyms and Symbols

Here, important acronyms, abbreviations, and symbols are listed, which are recurring throughout the thesis.

ACRONYMS AND ABBREVIATIONS

AI	Artificial Intelligence
ARR	Analytical Redundancy Relations
CCS	Cloud Control System
CIA	Confidentiality, Integrity, and Availability
CPS	Cyber-Physical System
DoS	Denial-of-Service
EMS	Energy Management System
IAE	Integral Absolute Error
ICS	Industrial Control System
IoT	Internet of Things
K8S	Kubernetes
LQR	Linear–quadratic regulator
ML	Machine Learning
MPC	Model Predictive Control
NCS	Networked Control Systems

PLC	Programmable Logic Controllers
PSO	particle swarm optimization
RTT	Round-Trip Time
SCADA	Supervisory Control and Data Acquisition
SVM	Support Vector Machine

INTRODUCTION

1 Introduction

Nowadays, computers have penetrated every aspect of our society, and constitute main parts of our daily life. The integration of the physical world with digital devices like computers has led to the emergence of a new generation of engineered systems: Cyber-Physical System (**CPS**). In these systems, embedded computers allow us to add capabilities to physical systems that we could not feasibly add in any other way [1].

An important example of **CPS** is Industrial Control System (**ICS**)s. **ICS**s are significant parts of the manufacturing industry and they play an important role in monitoring and controlling industrial processes [2]. We have had several revolutions in industry but in the middle of the twentieth century, computers and Programmable Logic Controllers (**PLC**) allowed us to have a higher level of automation. By combining computers and **PLC**s with the physical world, the existing industrial systems started becoming digitalized. This change with the digitalization of industry has been called Industry 3.0. First, the **PLC** replaced electric switches. Later, the **PLC**s gained increased sophistication with network capability. By merging computing and communication with physical processes and mediating the way we interact with the physical world, cyber-physical systems bring many benefits: they make systems safer and more efficient; they reduce the cost of building and operating these systems; and they allow individual machines to work together to form complex systems that provide new capabilities.

Today, the forth revolution, Industry 4.0, is happening, and will introduce intelligent manufacturing. Industry 4.0 constitutes a smart world of interconnected machines and intelligent robots that can talk to each other to share information, allowing for advanced analytics and visualizations of real-time data from multiple sources. Industry 4.0 is a step towards more automated systems coupled with computing devices, and will enable optimization of production flows and more efficient production [3]. Much of this is achieved with increased connectivity between different parts of the system. The increased connectivity in industrial control systems, together with the proliferation of connected cyber-physical systems, has led to more systems being connected to the Internet. While connectivity provides many advantages, it also opens systems to cyber attacks, and **ICS**s that have been traditionally isolated from the Internet, and thereby secured from various Internet-based security threats, will be vulnerable to remote attacks.

The connectivity will produce huge volumes of data that increases the communication and computational load, and it is not feasible to perform the processing on embedded devices. Therefore, one important underlying technologies in Industry 4.0 is the cloud. The cloud will provide enough computing power and resource storage, and the cloud is able to process big data, using Artificial Intelligence (**AI**) and machine learning methods. Also, the cloud makes it feasible to move the core processing of industrial control systems, i.e the controller, to the cloud and introduce Cloud Control System (**CCS**).

In **CCS**s, the core processing unit is shifted to a cloud server, which enables the control system to have massive parallel computation, and smaller size [4]. Moreover, this will lead to saved energy, by reducing the processing energy used in **ICS**s to enhance the system's lifetime. However, the connectivity in **CCS**s will also introduce new cyber security challenges. In these system, massages have to be sent from the physical systems to the controller in the cloud using a communication network, and this communication can be exposed to different types of security attacks. Recent attacks in different parts of the industry also demonstrate the necessity for an appropriate security measure to protect these systems.

In this thesis, we will address some security aspects of cloud-based industrial control systems, and our aim is to make these system attack resilient. This means that if the attacker infiltrates the system and establishes an attack, the system is able to detect this attack in a timely manner and mitigate its impacts on the system. So, the system will be resilient to the attack and can stay stable with an acceptable performance during the attack. Hence, we will suggest attack detection methods that continuously monitor the system for anomalies caused by adversary actions. Also, we propose some mitigation actions, which once an attack has been detected, reduce the impacts of the attack on the system.

1.1 OUTLINE

This thesis is structured as follows; Chapter 2 describes the general concepts related to the research in this thesis. Section 2.1 and 2.2 present an overview on Industry 4.0 and the cyber security challenges that are defined for these systems. Section 2.3 describes **CCS**s and how they will be the next version of networked control systems in Industry 4.0. Section 2.4 provides our vision on the different security aspects of control systems and studies how an attack can occur on these systems. Section 2.5 gives an overview of the experimental testbed, which is used as a proof-of-concept for evaluating our proposed solutions in our research. Chapter 4 provides a conclusion on how the work in each paper continued from the previous one, as well as our planned work following the research scope presented in this thesis. The second part of the thesis contains the publications.

2 Background

In this chapter, we introduce the general concepts related to the research including cloud control systems, cyber attacks, and different actions to protect these systems from cyber attacks. Also, we present a high level description on the experimental testbed, which we use as a proof-of-concept for evaluating our proposed solutions.

2.1 INDUSTRY 4.0

We have had several industrial revolutions. Figure 2.1 depicts these revolutions in a timeline. For many years and centuries, the production of goods and services such as food, clothing, etc., were manually performed by humans. However, with the beginning of the eighteenth century, these processes changed with the introduction of the first industrial revolution, and this period became a turning point in the industry. Steam and water power and mechanization in the early eighteenth century led to the beginning of Industry 1.0. During this period, mechanization led to an eightfold increase in production in the spinning industry. Steam as the main feature of this revolution in large industries led to the development and improvement of productivity. In this revolution, steam power replaced human muscular force in the spinning industry. Following the use of steam power, improvements were made in steam ships and steam locomotives and caused another great change in this period, since this technology reduced time for people and goods to travel long distances [5].

In the 19th century, electricity was discovered, and introduced as the main source of power. One of the advantages of electricity compared to steam and water was its ease of use, which enabled businesses to concentrate power

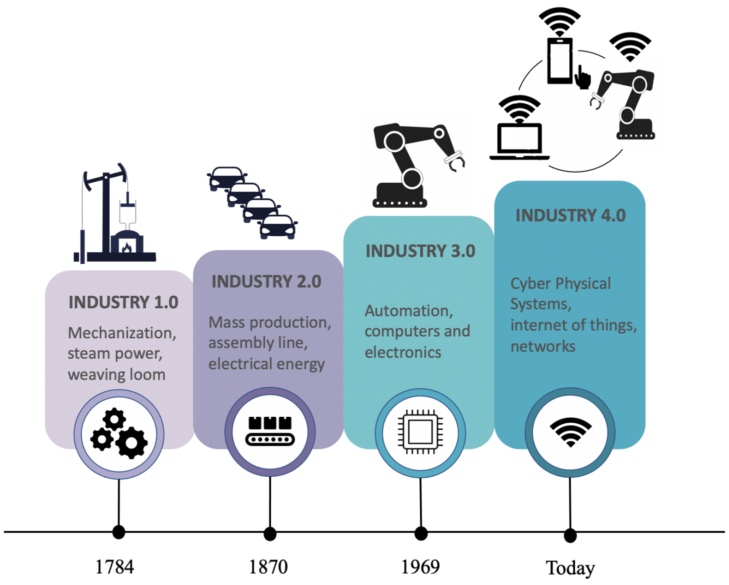

INDUSTRY 1.0
Mechanization, steam power, weaving loom
INDUSTRY 2.0
Mass production, assembly line, electrical energy
INDUSTRY 3.0
Automation, computers and electronics
INDUSTRY 4.0
Cyber Physical Systems, internet of things, networks
1784
1870
1969
Today

Figure 2.1: From Industry 1.0 to Industry 4.0.

sources to individual machines. Eventually, machines were designed with their own power sources, making them more portable. This period also saw the development of a number of management programs that made it possible to increase the efficiency and effectiveness of manufacturing facilities. Division of labor, where each worker does a part of the total job, increased productivity. Mass production of goods using assembly lines became commonplace. Finally, at the beginning of the 20th century the inventions of the automobile and the plane completed the second industrial revolution, or Industry 2.0 [6]. By the advent of electronics and computer technology in the middle of the twentieth century, Industry 3.0 was created. Two major inventions, Programmable Logic Controllers (**PLC**)s and Robots, helped give rise to an era of high-level automation.

Now, at the dawn of the 21st century, the world is witnessing the fourth industrial revolution, Industry 4.0, which is revolutionizing the way companies manufacture, improve and distribute their products. As Figure 2.2 shows, manufacturers are integrating new technologies, including Internet of Things (**IoT**), cloud computing, data analytics, and Artificial Intelligence (**AI**) and machine learning into their production facilities and throughout their operations [7]. These smart factories are equipped with advanced sensors, embedded software and robotics that collect and analyze data, and thereby allow for better decision making. Even higher business values are created

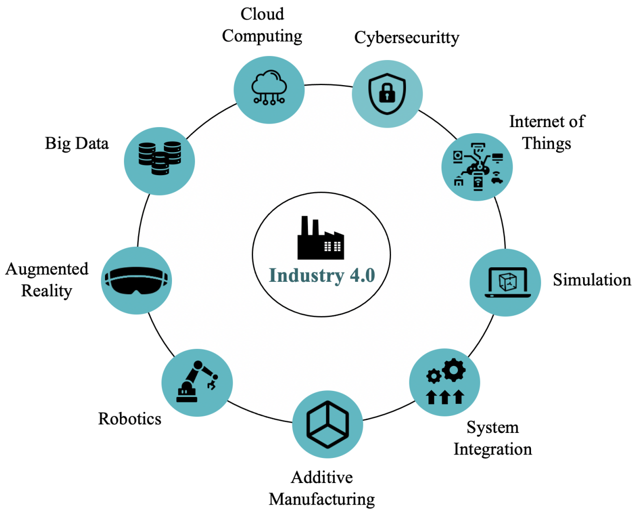


Figure 2.2: Main technologies of Industry 4.0.

when data from production operations is combined with operational data from supply chains, customer service and other enterprise systems, in order to create whole new levels of visibility and insights from previously siloed information systems.

These digital technologies lead to increased automation, predictive maintenance, self-optimization of process improvements and, above all, a new level of efficiencies and responsiveness to customers not previously possible. Developing smart factories provide an incredible opportunity for the manufacturing industry to enter the Industry 4.0. Analyzing large amounts of big data collected from sensors on the factory floor ensure real-time visibility of manufacturing assets, and can provide tools for performing predictive maintenance in order to minimize equipment downtime. Using high-tech **IoT** devices in smart factories lead to higher productivity and improved quality. Replacing manual inspection business models with **AI**-powered visual insights reduce manufacturing errors and saves money and time. With minimal investments, quality control personnel can set up a smartphone connected to the cloud to monitor manufacturing processes from virtually anywhere. By applying machine learning algorithms, manufacturers can detect errors immediately, rather than at later stages when repair work is more expensive. Industry 4.0 concepts and technologies can be applied across all types of industrial companies, including discrete and process manufacturing, as well as oil and gas, mining and other industrial segments [8].

2.2 CYBER SECURITY CHALLENGES IN INDUSTRY 4.0

The increased connectivity of smart machines in Industry 4.0, raises the stakes. Industry 4.0 heralds a new age of connected, smart manufacturing, responsive supply networks, and tailored products and services. Through its use of smart, autonomous technologies, Industry 4.0 strives to combine the digital world with physical actions, to drive smart factories and enable advanced manufacturing. But while it plans to enhance digital capabilities throughout the manufacturing and supply chain processes, and drive revolutionary changes to connected devices, it also brings with it new cyber risks for which the industry is unprepared [9]. Developing a fully integrated strategic approach to cyber risks is fundamental for smart manufacturing. As threats radically increase with the advent of Industry 4.0, new risks should be considered and addressed. Put simply, the challenge of implementing a secure, vigilant, and resilient cyber risk strategy is different in the age of Industry 4.0. When supply chains, factories, customers, and operations are connected, the risks posed by cyber threats become all the greater and potentially farther reaching [10].

2.3 CLOUD CONTROL SYSTEMS

Along with industrial revolutions, there has been many advances in network technologies, and these technologies have been combined with control systems and created Networked Control Systems (**NCS**). In this type of control systems, the control loop is closed via a communication channel, which makes it feasible to monitor and adjust the plant remotely. This integration of control systems and networks have increased the efficiency and agility of process control, and enabled more complex systems, such as Supervisory Control and Data Acquisition (**SCADA**).

NCSs have incorporated several functionalities, including reduced size, speed, and the ability to work for a long time. In turn, these functionalities demand that **NCS**s possess huge flexible computational resources of smaller size, which are difficult to achieve with the conventional design of these systems. Furthermore, control systems usually have to deal with big data, and this increases the communication and computational load of the network, which require high quality and real-time control to go beyond the traditional network control topology capability. Hence, by combining the benefits of network control and cloud computing technology, a new concept called Cloud Control System (**CCS**) has been developed, which can solve the issues of resource constrained **NCS**s. In **CCS**s, the core processing unit is shifted to a cloud server, which enables the control system to have massive parallel

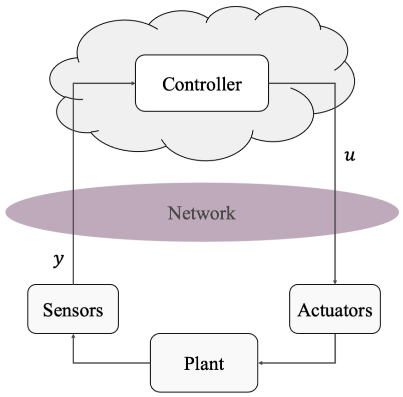


Figure 2.3: Cloud Control Systems overview.

computation, and also smaller size [11]. Moreover, using **CCS**s will lead to saved energy by reducing the processing energy used in **NCS**s, in order to enhance the system's lifetime.

Although there are many benefits of combining the cloud with control systems, this also leads to many security challenges. Figure 2.3 shows the general structure of a **CCS**. A **CCS** is composed of two layers: the cyber layer and the physical layer. The cyber layer consists of a communication network and a cloud, while the physical layer contains the plant, actuators and sensors [12]. Measurement signals y are sent from sensors in the physical domain to the controller in the cloud. The controller uses these measurement signals to generate the control signal u, which is sent back to the actuators in the physical domain. So, the controller in the cloud server, and the sensors in the physical domain are supposed to send packets through the communication channel. The communication channel can be exposed to different types of security attacks including passive and active attacks.

In recent years, there have been a number of attacks that targeted networked control systems and caused damage such as the Stuxnet attacks on Iran's nuclear installation in 2010 [13], the BlackEnergy malware attack on the Ukrainian power grid in 2015 [14], the HatMan malware attack on critical infrastructure using Schneider Electrics Safety Instrumented Systems in 2017 [15], the malware attack on 40 percent of all Industrial Control System (**ICS**) in energy organizations protected by Kaspersky Lab solutions in 2017 [16], and the cyber attacks on three U.S. natural gas pipeline companies in 2018 [17]. This indicates the possibility of such attacks on **CCS**s and highlights the need for appropriate security measures to protect these systems.

2.4 SECURITY ASPECTS OF CLOUD CONTROL SYSTEMS

In this section, different security aspects of control systems are investigated. Also, we describe how an attack can occur on theses system.

2.4.1 A CLOUD CONTROL SYSTEM UNDER ATTACK

Figure 2.4 shows a **CCS** under attack. The plant operation is supported by the communication network, through which the sensor measurements and actuator data are transmitted. On the plant side, the measurement data corresponds to $y_k \in \mathbb{R}^{n_y}$, while $\tilde{u}_k \in \mathbb{R}^{n_x}$ represents the actuator data. On the controller side, the sensor and actuator data are denoted by $\tilde{y}_k \in \mathbb{R}^{n_y}$ and $u_k \in \mathbb{R}^{n_n}$, respectively. The dynamical model of the plant is given by

$$\begin{cases} x_{k+1} = f_x\left(x_k, \tilde{u}_k, d_k\right) \\ \quad y_k = g_x\left(x_k, \tilde{u}_{k_r} d_k\right) \end{cases} \tag{2.1}$$

where $x_k \in \mathbb{R}^{n_x}$ denotes the plant's state, and the unknown input $d_k \in \mathbb{R}^{nd}$ models possible disturbances or faults affecting the system.

In normal condition we have: $\tilde{u}_k = u_k, \quad \widetilde{y}_k = y_k$, so mismatches between the transmitted signals and the received ones can be caused by the adversary's actions. For instance, an attacker that manipulates the measurement signals that are received by the controller can deceives the controller into generating the wrong control signal, which can make the plant unstable and cause damages. In real systems, due to the uncertainties and noise, transmitted

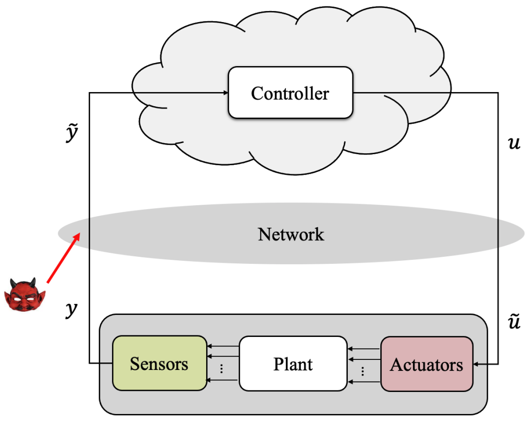


Figure 2.4: Cloud Control Systems under attack.

and received signals are not exactly the same, but they are close. Hence, for comparing these signals, a threshold is usually considered.

2.4.2 SAFE SET

In order to determine how an attack can affect a physical system and jeopardize it, we need to characterize the safety constraints of the system. For this, we use the safe sets concept based on [18]. Usually, physical systems have tight operating constraints that, if not satisfied, might result in physical damage to the system. For example, in power systems, cables cannot sustain an arbitrarily large instantaneous power. The system is said to be safe over the time interval $[0, N]$ if the state trajectory $\mathbf{x} \triangleq \left[x_0^\top \dots x_N^\top\right]^\top \in \mathbb{R}^{n_x(N+1)}$ is contained in the safe set $\mathcal{S}_x$. So based on these limitations by appropriate scaling of the output of the system y_k using λ, a safe set can be defined for this system as follows:

$$\mathcal{S}_x = \left\{ \mathbf{x} : \max_k \{\|C_x x_k\|_\infty\} \leq \lambda \right\} \tag{2.2}$$

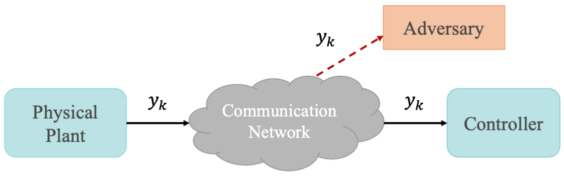
The system is said to be safe if the state trajectory x_k remains in S_x. Therefore, the attacker that wants to damage the system tries to drive the state of the system out of its safe set.

2.4.3 CYBER ATTACKS ON CLOUD CONTROL SYSTEMS

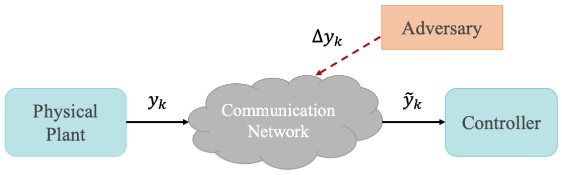
To model and understand how a cyber adversary may affect the cloud control system's operation requires knowing how IT systems are vulnerable to adversaries. The computer security literature identifies three fundamental properties of information and services in IT systems, namely confidentiality, integrity, and availability, often denoted as **CIA** [19], and they can be violated by disclosure, deception, and denial-of-service attacks, respectively.

Confidentiality concerns the concealment of data, ensuring it remains known only to the authorized parties. However, in a disclosure attack, as can be seen in Figure 2.5a, the attacker intrudes the telecommunication network and eavesdrops on a transmitted message. Although this kind of attack does not have a devastating effect on the system, the attacker may use it to get some knowledge about the system and launch more complicated attacks [20].

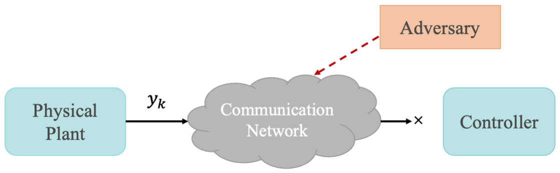
Integrity relates to the trustworthiness of data, meaning there is no unauthorized change to the information between the source and destination. In a deception attack, the attacker manipulates the data that is sent through the network, as is seen in Figure 2.5b. For example, by injecting false data to the measurement signal that is sent to the controller, the data integrity is violated and the controller is deceived to generate the wrong control signal.



(a) Disclosure attack.



(b) Deception attack.



(c) DoS attack.

Figure 2.5: Cyber attacks on a communication network

Availability considers the timely access to information or system functionalities. In a Denial-of-Service (**DoS**) attack, the attacker occupies the network bandwidth, and thereby prevents the message from arriving at the destination [21]. As can be seen in Figure 2.5c, the message sent by the plant is actually blocked and does not reach the controller.

In this research, we consider deception attacks where the attacker manipulates the real value of one or several signals, similar to the Stuxnet attack where malware manipulated the speed of centrifuges in a nuclear enrichment plant, and caused them to spin out of control. So, we consider deception attacks where the attacker tries to manipulate the data integrity for the transmitted packets between different components of the cyber physical system. So, in the **CCS** case, the attacker may manipulate the measurement signal y or the control signal u in Figure 2.4. For instance, the attacker may add an attack vector $f_a = [a_1 \quad a_2 \quad ... \quad a_p]^T$ to the measurement signal $y = [y_1 \quad y_2 \quad ... \quad y_p]$, and this attack vector has nonzero entries for measurements under attack and zero values for all other measurements. This means that we can model a deception attack as follows:

$$\tilde{y}_k = y_k + f_a \tag{2.3}$$

The controller will receive the manipulated measurement signal(s) instead of the real one(s), and based on these manipulated signals generate a control signal that can damage the system. In the next section, we describe the general solutions for this deception attacks.

2.4.4 DIFFERENT ACTIONS TO PROTECT ICS FROM CYBER ATTACKS

The different actions to protect **ICS**s from cyber attacks can be classified as

- Prevention
- Detection
- Mitigation

Prevention aims at decreasing the likelihood of attacks by reducing the vulnerability of the system components, for instance by encrypting the communication channels, using firewalls, and security protocols [22]. On the other hand, detection is an approach in which the system is continuously monitored for anomalies caused by adversary actions, and once an attack is detected, mitigation actions try to reduce the impacts of the attack on the system [18].

There are two important reasons for why detection and mitigation actions are necessary, and why only prevention actions like encryption is not enough. First of all, there can come a powerful attacker who can break the prevention actions and intrude into the system to establish a malicious attack. In recent years, there have been several attacks in different parts of industry, which show that there have some attackers who could break the prevention layer and infiltrate the system. So, detection and mitigation actions are needed to make the system able to tolerate attacks and remain stable. Second, in some systems for example power grids, most parts of the equipment are old, and implementing prevention measures like encryption will be costly, because of the required updates of the equipment. Therefore, in this case detection and mitigation actions can be used that are completely adaptable to already implemented industrial control systems. Also, given limited protection resources (the number of devices for data encryption), prevention methods can be combined with detection and mitigation actions, in order to find which signals that should be encrypted in order to maximize the benefits of the protection resources. The main aim in our research is to design methods related to detection and mitigation actions.

2.4.5 DIFFERENCE BETWEEN FAULT AND ATTACK

There have been many studies on fault-tolerant control, for example [23–25], which can provide tools for attack-resilient control systems as well. However, there are substantial differences between the fault-tolerant and attack-resilient control when it comes to attack detection and mitigation, which motivate the need for specific methodologies to address security issues in **ICS**s. For example, faults are considered as physical events that affect the system behavior, where the events do not act in a coordinated way, while cyber attacks may be performed over a significant number of attack points in a coordinated fashion [18,26]. In addition, faults do not have an intent or objective to fulfill, while cyber attacks have a malicious intent.

2.5 TESTBED DESCRIPTION

As a proof-of-concept for evaluating the proposed solutions in our research, we have evaluated them on an experimental testbed, as shown in Figure 2.6. This section describes the testbeds different parts.

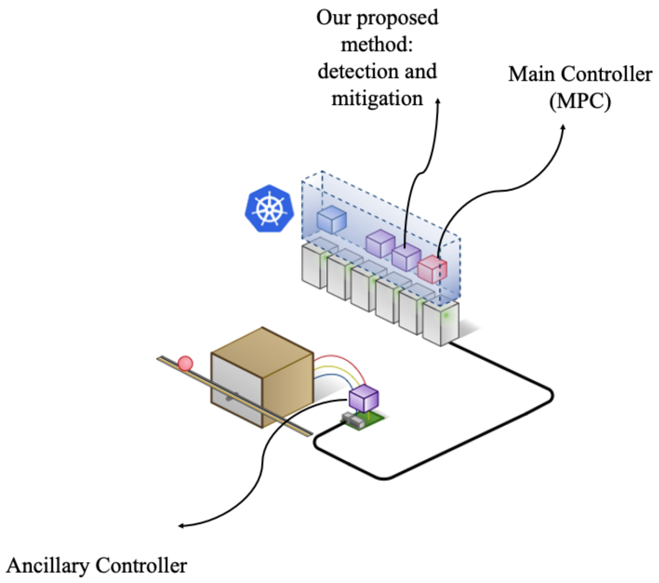


Figure 2.6: testbed overview.

2.5.1 PLANT

In the testbed, the ball and beam process is used as plant. The ball and beam system includes a long beam, where the ball is rolling back and forth. This system is open-loop unstable and the ball can swing and fall off the end of the beam. So, the controller tries to hold the ball on the setpoint on top of the beam by tilting the beam using an electrical motor. We define a safe set, as described in Section 2.4.2, for the ball and beam system based on the length of the beam. Since the length of the beam is 1.1 m, the allowed range for the position of the ball is $[-0.55m, 0.55m]$. The attacker's goal is to drive the ball out of this range, and thereby cause the ball to fall off the end of the beam. Also, if the attacker moves the ball from its predefined setpoint but hold it on the beam, it may not damage the system, but can cause extra cost and decrease the efficiency. Hence, the aim of the proposed methods in this thesis is to hold the ball not only on the beam, but also on the exact setpoint. We have chosen the ball and beam system as a plant, since it has fast dynamics and it is a time critical system. Also, even in the absent of an attack, controlling the system over the cloud is tricky. Hence, by applying our proposed methods for this process, and keeping it stable in the presence of attacks, will prove the effectiveness of our methods very well.

The ball and beam system has three measurement signals: the position of the ball y_1, the speed of the ball y_2, and the angle of the beam y_3. The process can be modeled in continuous time as follows:

$$\begin{aligned}\dot{x}(t) &= \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & -\frac{5g}{7} \\ 0 & 0 & 0 \end{bmatrix} x(t) + \begin{bmatrix} 0 \\ 0 \\ 0.44 \end{bmatrix} u(t) \\ y(t) &= \begin{bmatrix} a_1 & 0 & 0 \\ 0 & b_2 & 0 \\ 0 & 0 & c_3 \end{bmatrix} x(t)\end{aligned} \tag{2.4}$$

where $g = 9.80665$ is the gravity of Earth. We discretize this continuous time model with a sampling time 0.05 s for designing the controller and our security framework.

2.5.2 KUBERNETES CLUSTER

The testbed includes a seven-node Kubernetes cluster, used as an edge cloud. Kubernetes (**K8S**) is a portable, extensible, open-source platform for managing containerized workloads and services, that facilitates both declarative configuration and automation [27]. The cluster has been equipped with an nginx ingress [28] and Prometheus operator [29]. The nginx ingress is exposed using the **K8S** NodePort paradigm. We use this **K8S** cluster to implement our main controller and attack detection algorithm.

2.5.3 MAIN CONTROLLER

To stabilize the ball, we need a feedback controller that uses measurement signals to adjust the beam accordingly. An Model Predictive Control (**MPC**)-based controller was designed based on [30] as the main controller in Figure 2.6. **MPC** is an advanced method of process control, which is used to control a process while satisfying a set of constraints. **MPC** relies on a dynamic model of the process and uses this model to forecast the system behaviour. Hence, the main advantage of this controller is the fact that it allows the current time slot to be optimized, while taking future time slots into account. The current control action is obtained by solving, at each sampling instant k, a finite horizon (N) open-loop optimal control problem, using the current state of the plant as the initial state as follows:

$$\begin{aligned} \underset{\mathbf{u}}{\text{minimize}} J &= \sum_{i=k}^{k+N-1} x_i^T Q x_i + u_i^T R u_i + x_{k+N}^T P x_{k+N} \\ \text{subjectto} \quad x_{i+1} &= A x_i + B u_i \\ G\begin{bmatrix} x_i \\ u_i \end{bmatrix} &\leq g, \quad H\begin{bmatrix} x_i \\ u_i \end{bmatrix} = h \quad , x_{n+k} \in \mathcal{T} \end{aligned} \tag{2.5}$$

where Q, R and P are cost matrices, A and B define the model of the system, x is the state vector, u is the control signal, and the constraints of the system are defined by the matrices and vectors G, g, H and h. This optimization yields an optimal control sequence $u(k), u(k+1), ..., u(k+N-1)$ and the first control in this sequence $u(k)$ is applied to the plant.

As is explained in Section 2.5.1, controlling the ball and beam system over the cloud may fail due to its fast dynamics and the delay between the cloud and the plant. Since **MPC** is implemented in the cloud, in this research, in order to compensate the possible network delay between the plant and the controller in the **K8S** cluster, we apply the future control sequence $u(k+m)$ instead of the first control value in this sequence $u(k)$. Actually, at each sampling instant k, we measure the Round-Trip Time (**RTT**), i.e. the time it takes to send the measurement signal and receive the control signal. By dividing the **RTT** with the sampling time, we can calculate m, and decide which step of the control sequence $u(k+m)$ to use instead of the first step $u(k)$. For example, in Figure 2.7, the **RTT** is more than one sampling time (h), which means that, for instance, the control signal related to y_0 that should be received by plant between 0 and 1, instead is received between 1 and 2. So, in

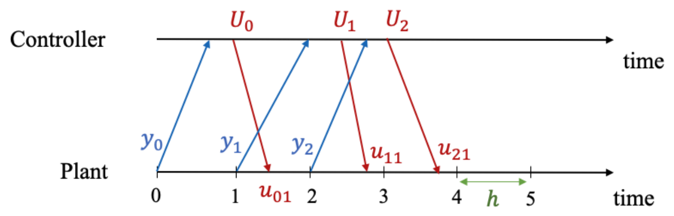


Figure 2.7: Delay compensation using MPC.

order to compensate for this delay, we apply the second control signal in the control sequence $U_0 = \{u_{00}, u_{01}, ..., u_{0N-1}\}$ that is related to the next sampling time.

In this test-bed, **MPC** is implemented using Python and container technology, and it is deployed to the **K8S** cluster as a pod.

2.5.4 ANCILLARY CONTROLLER

In some parts of our research, we need to implement a local controller in the physical domain. Hence, this testbed has been equipped with a local controller that is implemented as an Linear–quadratic regulator (**LQR**). The control signal of the **LQR** is generated as follows:

$$u_k = -Kx_k \tag{2.6}$$

where K is the gain vector and x is the state vector. **LQR** is a simpler controller than **MPC** and requires less computational capacity, allowing it to be implemented in the physical domain. This controller is implemented close to the plant. This means that there is no network between the **LQR** and the plant, which means that the controller is secure. This means that an attacker does not have any access to the measurement signals that are sent to the **LQR**. The controller receives the measurement signal and works in parallel with the **MPC** controller, but the priority is to use the control signal of the **MPC**. Only in some special conditions, the system can switch to the local controller.

3 Summary and Contributions

3.1 RESEARCH CONTRIBUTIONS

The four papers included in this thesis are summarized below, which illustrates the path of our investigation on security of industrial cloud control systems. This chapter gives an overview on the content of each paper, and detail my main contributions in each work.

In this research, we have investigated the security challenges for cloud-based industrial control systems, Cloud Control System (**CCS**)s. We started by proposing methods for detecting attacks in **CCS**s, and evaluated these methods in simulations in Matlab (paper I). Then we proposed a security framework by equipping the detection method with a mitigation method, in order to keep the plant stable with an acceptable performance during an attack. We showed the performance of this framework by implementing it on a real testbed (paper II and III). Finally, we improved the mitigation part of the proposed security framework in our previous papers, and instead of using an ancillary controller during the attack, we proposed reconfiguration of the main controller, such that it can tolerate an attack. We also showed the performance of this solution by implementing it on a real testbed (paper IV).

3.1.1 PAPER I: INTRUSION DETECTION IN DIGITAL TWINS FOR INDUSTRIAL CONTROL SYSTEMS

In this paper, we propose implementing a digital twin for **CCS**s, which can be equipped with an intrusion detection algorithm. Our proposed intrusion algorithm is able to detect attacks in a timely manner and also diagnose the type of attack by classification of different types of attacks. With a digital twin, which is a new concept in Industrial Control System (**ICS**)s, there is a virtual replica of the physical systems, which can precisely mirror the internal behavior of the physical systems. So by placing the intrusion detection algorithm in the digital twin, security tests can be done remotely without risking negative impacts on the live system. In this paper, we assume that the attacker intrudes into the system through the communication network and manipulates the measurement signals that are sent from the plant to the controller, in order to deceive the controller into generating a wrong control signal, which can make the system unstable and cause damages. For this, we propose two different knowledge-based and data-driven attack detection methods. In the knowledge-based attack detection method, by designing a Kalman filter, the measurement signals are estimated, and by comparing this estimated signal with the real measurement signals, the attack can be detected. In the data-driven attack detection method, a Support Vector Machine (**SVM**), which is a machine learning approach, is used for both attack detection and classification. The **SVM** method is able to diagnose if there is an attack in the system or not, and if an attack is detected, diagnose the type of this attack. We evaluate our proposed approach in Matlab.

Contribution: I was the primary researcher in this work and my contribution is in system definition and modelling, solution design, simulation development, performance evaluation and analysis.

3.1.2 PAPER II: A SECURITY FRAMEWORK IN DIGITAL TWINS FOR CLOUD-BASED INDUSTRIAL CONTROL SYSTEMS: INTRUSION DETECTION AND MITIGATION

In this paper, we propose a digital twin-based security framework for cloud-based industrial control systems, which consists of two parts: attack detection and attack mitigation. In order to detect attacks, a residual signal is generated, which in the absence of any attack has a nominal value close to zero and may deviate a little bit from zero only due to modelling uncertainties and noise. However, if an attack occurs, the residual deviates from zero with a magnitude such that the new condition can be distinguished from the attack-free mode. For this, we employ a Kalman filter to estimate the behaviour of the system and then by comparing it with the real measurement signals, the residual signal is generated and used for detecting if the system is in a normal condition or if there is an attack in the system. As mitigation part, we employ a local controller in addition to the cloud-based controller. The local controller is in the physical domain close to the plant, such that there is no public network between this controller and the plant. Consequently, the local controller is secure, and there is no possibility of intrusion into the system by attackers. Measurement signals are sent from the sensors to both controllers and they both generate control signals, but the priority is to use the control signal generated by the controller in the cloud, since this is a more advanced controller. Once an attack has been detected, an alarm signal is triggered and makes the system able to switch to the local controller, and instead use the control signal generated by this controller, since the cloud domain of the system is not secure and signals from the cloud cannot be trusted. We implement our framework on the experimental testbed and evaluate its capability by subjecting it to a set of attacks. We show that our proposed framework can detect attacks in a timely manner and keep the system stable with acceptable performance during the attack.

Contribution: I was the primary researcher in this work and my contribution is in system definition and modelling, solution design, simulation development, experiment setup, performance evaluation and analysis.

3.1.3 PAPER III: A CLOUD-CONTROL SYSTEM EQUIPPED WITH INTRUSION DETECTION AND MITIGATION

In this paper, we present a demo of the security framework that we proposed in paper II for cloud-based industrial control system, and demonstrate how it can detect attacks on this system quickly and mitigate them. Also, in this paper we explain different parts of our experimental testbed in detail, and we show how our algorithm is applied on this testbed.

Contribution: I was the primary researcher in this work and my contribution is in system definition and modelling, solution design, simulation development, experiment setup, performance evaluation and analysis.

3.1.4 PAPER IV: ATTACK RESILIENT CLOUD CONTROL SYSTEMS

In this paper, we propose a security framework for cloud-based industrial control systems to make them resilient against attacks. This framework includes three steps: attack detection, attack isolation,and attack mitigation. In the attack detection part, a residual signal is generated such that it is close to zero and less than a predefined threshold in normal condition during which there is no attack, but will exceed the threshold once the attack has occurred. We investigate two different observer-based and Analytical Redundancy Relations (**ARR**) methods to generate residual signals and compare their efficiency to detect attacks. In the mitigation part, once the attack has been detected, the controller is reconfigured such that it can tolerate the attack. This reconfiguration is done by developing a virtual sensor concept, which is a method used to deal with sensor failures. In this mitigation method, isolation is the necessary and main part that is used to know exactly on which sensor(s) there is an attack. So, we propose a novel isolation method that can exactly diagnose on which sensor(s) an attack has occurred, and then we use this information in mitigation part. We also compare our proposed isolation method with the common available method for isolation, the **ARR** method. We show defects of the **ARR** method, and we show how our method is more powerful than **ARR** to diagnose accurate locations of attacks. We validate our proposed framework on the experimental testbed, and evaluate its capability by subjecting it to a set of attacks. We show that our proposed solution can detect the attack in a timely manner, and that it can keep the plant stable with an acceptable performance during the attack.

Contribution: I was the primary researcher in this work and my contribution is in system definition and modelling, solution design, simulation development, experiment setup, performance evaluation and analysis.

3.2 CONCLUSIONS AND FUTURE WORK

In the introduction of this thesis, we have walked through the development of the industrial control systems, which by combining the benefits of network control and cloud computing technology, a new concept called cloud control system has been developed that can almost solve issues of resource constrained Networked Control Systems (**NCS**)s. Also by shifting core processing units to cloud servers, control system is able to have massive parallel computation, and also smaller size. Moreover, this will lead to saved energy by reducing the processing energy used in **NCS**s to enhance the system's lifetime.

However, the cloud introduces major security challenges, since moving control systems to the cloud can enable attackers to infiltrate the system and establish an attack, which can lead to damages and disruptions with potentially catastrophic consequences. Various measures to protect these systems against cyber attacks can be classified as prevention, detection and mitigation. In our research, we investigated and designed methods related to detection and mitigation actions, in order to detect and mitigate deception attacks on cloud-based industrial control system.

With the growth of advances in technologies, attacks are also becoming more complex and attackers, by earning some knowledge about the systems, are becoming able to design optimal deception attacks that can cause biggest damage with lower possibilities of detection. Hence, as future work, we will expand our proposed framework for such attacks.

Also, except the deception attack that we considered in our research, there are other kinds of attacks that should be considered. In our ongoing work, we are designing some methods for detecting replay attacks on cloud control systems. In this kind of attack, the attacker can, by performing a disclosure attack, gather a sequence of measurement signals, and can then begin to replace true measurement of the systems by replaying the previously recorded data in order to remain stealthy. In this scenario, the attacker is also able to perform a physical attack while replaying the recorded data.

Moreover, we will also expand our work to other challenges of cloud-based industrial control systems, for example delay compensation. Moving core processing unit to a cloud server will lead to some delay in reaching the control signal to the plant, which can make the system unstable. Hence, we will design methods that can make the system able to work well in presence of this delay.

Bibliography

[1] L. Monostori, B. Kádár, T. Bauernhansl, S. Kondoh, S. Kumara, G. Reinhart, O. Sauer, G. Schuh, W. Sihn, and K. Ueda, "Cyber-physical systems in manufacturing," *Cirp Annals*, vol. 65, no. 2, pp. 621–641, 2016.

[2] K. Stouffer, S. Lightman, V. Pillitteri, M. Abrams, and A. Hahn, "Nist special publication 800-82, revision 2: Guide to industrial control systems (ics) security," *National Institute of Standards and Technology*, 2014.

[3] T. Devezas and A. Sarygulov, *Industry 4.0.* Springer, 2017.

[4] Y. Xia, Y. Qin, D.-H. Zhai, and S. Chai, "Further results on cloud control systems," *Science China Information Sciences*, vol. 59, no. 7, pp. 1–5, 2016.

[5] U. A. Pozdnyakova, V. V. Golikov, I. A. Peters, and I. A. Morozova, "Genesis of the revolutionary transition to industry 4.0 in the 21st century and overview of previous industrial revolutions," in *Industry 4.0: Industrial Revolution of the 21st Century*. Springer, 2019, pp. 11–19.

[6] Y. Yin, K. E. Stecke, and D. Li, "The evolution of production systems from industry 2.0 through industry 4.0," *International Journal of Production Research*, vol. 56, no. 1-2, pp. 848–861, 2018.

[7] L. S. Dalenogare, G. B. Benitez, N. F. Ayala, and A. G. Frank, "The expected contribution of industry 4.0 technologies for industrial performance," *International Journal of production economics*, vol. 204, pp. 383–394, 2018.

[8] K. Kumar, D. Zindani, and J. P. Davim, *Industry 4.0: developments towards the fourth industrial revolution*. Springer, 2019.

[9] R. Waslo, T. Lewis, R. Hajj, and R. Carton, "Industry 4.0 and cybersecurity: Managing risk in an age of connected production," *Deloitte Insights.*, 2017.

[10] M. M. Alani and M. Alloghani, “Security challenges in the industry 4.0 era,” in *Industry 4.0 and engineering for a sustainable future*. Springer, 2019, pp. 117–136.

[11] Y. Xia, “Cloud control systems,” *IEEE/CAA Journal of Automatica Sinica*, vol. 2, no. 2, pp. 134–142, 2015.

[12] Z. Xu and Q. Zhu, “Secure and resilient control design for cloud enabled networked control systems,” in *Proceedings of the first ACM workshop on cyber-physical systems-security and/or privacy*, 2015, pp. 31–42.

[13] R. Langner, “Stuxnet: Dissecting a cyberwarfare weapon,” *IEEE Security & Privacy*, vol. 9, no. 3, pp. 49–51, 2011.

[14] D. Alert, “Cyber-attack against ukrainian critical infrastructure,” *Cybersecurity Infrastruct. Secur. Agency, Washington, DC, USA, Tech. Rep. ICS Alert (IR-ALERT-H-16-056-01)*, 2016.

[15] ICS-CERT, “Hatman—safety system targeted malware,” Mar. 2017. [Online]. Available: https://ics-cert.us-cert.gov/MAR-17-352-01-HatManTargeted-Malware.

[16] Kaspersky Lab ICS-CERT, “Threat landscape for industrial automation systems in h2 2017,” Mar. 2018. [Online]. Available: https://icscert.kaspersky.com/reports/2018/03/26/threat-landscape-for-industrial-automation-systems-in-h2-2017/.

[17] N. S. Malik, R. Collins, and M. Vamburkar, “Cyber-attack, pings data systems of at least four gas networks,” Apr. 2018. [Online]. Available: https://www.bloomberg.com/news/articles/2018-04-03/day-after-cyberatta ck-a-third-gas-pipeline-data-system-shuts.

[18] A. Teixeira, K. C. Sou, H. Sandberg, and K. H. Johansson, “Secure control systems: A quantitative risk management approach,” *IEEE Control Systems Magazine*, vol. 35, no. 1, pp. 24–45, 2015.

[19] M. A. Bishop, “The art and science of computer security,” 2002.

[20] A. Teixeira, D. Pérez, H. Sandberg, and K. H. Johansson, “Attack models and scenarios for networked control systems,” in *Proceedings of the 1st international conference on High Confidence Networked Systems*, 2012, pp. 55–64.

[21] M. S. Chong, H. Sandberg, and A. M. Teixeira, “A tutorial introduction to security and privacy for cyber-physical systems,” in *2019 18th European Control Conference (ECC)*. IEEE, 2019, pp. 968–978.

[22] O. Vukovic, K. C. Sou, G. Dan, and H. Sandberg, “Network-aware mitigation of data integrity attacks on power system state estimation,” *IEEE Journal on Selected Areas in Communications*, vol. 30, no. 6, pp. 1108–1118, 2012.

[23] M. Blanke, M. Kinnaert, J. Lunze, M. Staroswiecki, and J. Schröder, *Diagnosis and fault-tolerant control*. Springer, 2006, vol. 2.

[24] Z. Gao, C. Cecati, and S. X. Ding, "A survey of fault diagnosis and fault-tolerant techniques—part i: Fault diagnosis with model-based and signal-based approaches," *IEEE transactions on industrial electronics*, vol. 62, no. 6, pp. 3757–3767, 2015.

[25] G. Zhiwei, C. Cecati, and S. X. Ding, "A survey of fault diagnosis and fault-tolerant techniques—part ii: Fault diagnosis with knowledge-based and hybrid/active approaches," 2015.

[26] A. Teixeira, G. Dán, H. Sandberg, and K. H. Johansson, "A cyber security study of a scada energy management system: Stealthy deception attacks on the state estimator," *IFAC Proceedings Volumes*, vol. 44, no. 1, pp. 11 271–11 277, 2011.

[27] Kubernetes. [Online]. Available: https://kubernetes.io

[28] Ingress-nginx. [Online]. Available: https://github.com/kubernetes/ingress-nginx

[29] Prometheus-operator. [Online]. Available: https://github.com/coreos/prometheus-operator

[30] P. Skarin, W. Tärneberg, K.-E. Årzén, and M. Kihl, "Control-over-the-cloud: A performance study for cloud-native, critical control systems," in *2020 IEEE/ACM 13th International Conference on Utility and Cloud Computing (UCC)*. IEEE, 2020, pp. 57–66.

PAPERS

Paper I

Paper I

Reproduced, with permission, from

<u>Fatemeh Akbarian</u>, Emma Fitzgerald, Maria Kihl, "Intrusion Detection in Digital Twins for Industrial Control Systems," *The 28th International Conference on Software, Telecommunications and Computer Networks (SoftCOM)*, Hvar, Croatia, Sept. 2020.

I

Intrusion Detection in Digital Twins for Industrial Control Systems

Nowadays, the growth of advanced technologies is paving the way for Industrial Control System (**ICS**) and making them more efficient and smarter. However, this makes **ICS** more connected to communication networks that provide a potential platform for attackers to intrude into the systems and cause damage and catastrophic consequences. In this paper, we propose implementing digital twins that have been equipped with an intrusion detection algorithm. Our novel algorithm is able to detect attacks in a timely manner and also diagnose the type of attack by classification of different types of attacks. With digital twins, which are a new concept in **ICS**, we have virtual replicas of physical systems so that they precisely mirror the internal behavior of the physical systems. So by placing the intrusion detection algorithm in digital twins, security tests can be done remotely without risking negative impacts on live systems.

1 INTRODUCTION

Today, smart manufacturing has attracted much attention [1], and the growth of Internet of Things (**IoT**) and cloud computing are paving the way for the smart factories that will realize Industry 4.0. As part of Industry 4.0, Industrial Control Systems (**ICS**), which consists of combinations of control components that act together to achieve an industrial objective, are becoming connected and part of the networked systems in the factory. Although, networked **ICS**s increase the efficiency of the systems, they may jeopardize the system's security at the same time, since they can provide a critical platform through which attackers may be able to intrude into the system.

Some examples of recent cyber attacks that targeted **ICS**s are the Stuxnet computer worm attacks on Iran's nuclear installation in 2010 [2], BlackEnergy malware attack on the Ukrainian power grid in 2015 [3], HatMan malware attack on critical infrastructure using Schneider Electrics Safety Instrumented Systems in 2017 [4], malware attack on 40 percent of all **ICS** in energy organizations protected by Kaspersky Lab solutions in 2017 [5], and the cyber attacks on three U.S. natural gas pipeline companies in 2018 [6]. These attacks demonstrate security weaknesses and the necessity for appropriate security measures to protect **ICS** infrastructure.

According to this significant increase in cyber attacks, much attention has been paid to intrusion detection in **ICS** over the recent years.

Digital twin is a rather new concept in industry. With digital twins, we have virtual replicas of physical systems so that they precisely mirror the internal behavior of the physical systems [7]. So using this virtual environment for security tests instead of the real system prevents any interference with the live systems.

Hence, in this paper, we propose a digital twin based intrusion detection technique due to a couple of reasons. First, applying security tests may have some negative effects on the live systems and cause to reduce the efficiency of the systems. Also, an intrusion detection system in the digital domain can include methods that require much more computing resources than if deployed in the real system. For instance, machine learning techniques usually are hard to realise in the physical domain, where there often are embedded devices with limited computing power and more restrictive programming models.

There are a few papers that have suggested intrusion detection as a use case for digital twins. To the best of our knowledge, so far, only two papers have been published that show how the concept of digital twins can be used to implement intrusion detection systems. Authors in [8], defined two rules, namely, safety and security rules, that specific digital twins must adhere to. However, this research missed the synchronization between digital twins and the real systems. So real systems' data is not incorporated into the implemented intrusion detection. Moreover, the proposed rules are very limited as an intrusion detection method for detecting cyber attacks. Regarding these issues, authors in [9] proposed a passive state replication to create synchronization between physical systems and digital twins that is a fundamental requirement for realizing the intrusion detection using digital twins. However, this synchronization method only copy some limited data of physical system to digital twin and it does not make the digital twin able to follow the physical system continuously for instance, when there are unexpected changes in the physical system. Further, the authors proposed a behavior-specification-based intrusion detection to determine whether the system's behavior during runtime diverges from the predefined correct behavior due to an intrusion. However, creating the specification of the system's correct behavior typically requires processing effort, whereas in this paper the authors sidestep this issue by making the assumption that the specification of the system is readily available.

Motivated by synchronization challenges in digital twins, in our prior work [10] we proposed an architecture to implement a digital twin that makes it able to follow its physical counterpart's behavior continuously and guarantees synchronization between digital twins and physical systems. In this paper, we equip this architecture with a novel intrusion detection algorithm that unlike [9] does not need a specification of the system's correct behavior. We demonstrate how the digital twin concept can be used for intrusion detection. Further, we develop a novel algorithm in the digital twin for timely detection of attacks on **ICS**. Also, we propose an approach to diagnose the type of attack after detecting it by classification of different types of attacks. Finally, we evaluate the capability of the proposed anomaly detection algorithm through simulation studies.

2 TARGETED SYSTEM

The targeted system, which concerns industrial control systems, is illustrated in Figure 1. In this figure, the physical domain shows an industrial system, which is composed of several different control systems. These control systems can represent a factory where there are different kinds of machines and robots. A digital twin for the system is deployed in the cloud using the architecture in our prior work [10]. There will always be a network between the physical domain and the cloud, and also there can be a network between the components of a control system inside the physical domain. These networks create the possibility of cyber attacks on the signals that are sent through them.

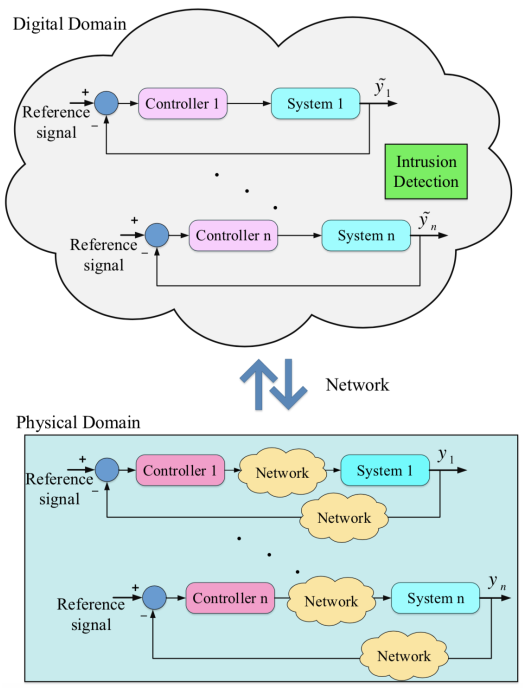


Figure 1: Targeted system overview.

In this paper, as Figure 1 shows, an intrusion detection system is deployed in the digital twin in order to detect cyber attacks on the real system. Also, we consider attacks that intrude into the network and try to manipulate the measurement signal y in order to drive the system to an unsafe state.

We consider two Scaling and Ramp attacks in which the attacker inject false data into the signal and modify it. According to [11], [12] and [13], we can model these attacks as follows:

- Scaling attack: In this type of attack, the measurement signal is manipulated and its value depending on the amount of scaling attack parameter λ_s, is converted to a value greater or less than the actual value:

$$y^*(t) = \begin{cases} y(t) & \text{for } t \notin \tau_a \\ (1+\lambda_s)\cdot y(t) & \text{for } t \in \tau_a \end{cases} \tag{1}$$

  where τ_a indicates the attack period.
- Ramp attack: In this type of attack, since the beginning of the attack, $\lambda_r.t$ is added to the actual signal and depending on the amount of λ_r,

increases or decreases the value of the signal:

$$y^*(t) = \begin{cases} y(t) & \text{for } t \notin \tau_a \\ y(t) + \lambda_r.t & \text{for } t \in \tau_a \end{cases} \tag{2}$$

Since different attacks can have different effects on the system, different mitigation methods may be needed to deal with them. Therefore, after detecting the attack, it is also important to classify the attack in order to choose the best mitigation method for it. Hence, in next section, we explain our proposed method to detect these attacks in a timely manner and classify them.

3 PROPOSED SOLUTION

In this paper, we propose a novel intrusion detection, and we also propose to apply this intrusion detection to digital twins instead of the real system. Our proposed solution consists of two parts: attack detection and attack classification.

3.1 ATTACK DETECTION

In order to detect an attack on the system, we propose to use a Kalman filter [14] to estimate the correct signals in the system. The Kalman filter uses input and output signals of the system to estimate the correct output of the system. This means that the Kalman filter optimally removes the destructive effects of the attack and noises from the manipulated signal and can estimate the correct behavior of the system. This estimated signal can then be used to detect the occurrence of the attack. In order to design a Kalman filter, first, an observable state-space model of the system is needed. The Kalman filter will be placed in the digital twin, and as the digital twin is created based on [10], we can assume that there exists a simulated model of the real system that has been obtained easily by system identification algorithms. Therefore, it is only necessary to create an observable realization of this model to generate an observable state-space model.

After designing the Kalman filter, input and output signals of the system, which, the Kalman filter has been designed for, should be sent to the Kalman filter to estimate the correct output signal.

By considering process noise and measurement noise in the system, our targeted system can be modelled as follows:

$$\begin{aligned} x_{\mathrm{k}+1} &= Ax_{\mathrm{k}} + Bu_{\mathrm{k}} + Gw_{\mathrm{k}} & w &\rightarrow N(0,Q) \\ y_{\mathrm{k}} &= Cx_{\mathrm{k}} + Fv_{\mathrm{k}} & v &\rightarrow N(0,R) \end{aligned} \tag{3}$$

In this model x is the state vector, y is the output signal which is measured by sensors, u is the input signal which is generated by the controller, w is process noise, v is measurement noise and subscript k shows time instance. Here, we consider process noise and measurement noise to be white noise with covariances Q and R respectively. Also, A, B, C, G, and F are coefficient matrices.

A Kalman filter for this system will be designed using the following recursive algorithm, which consists of two parts: time update and measurement update [14]. The time update part consists of the following steps:

$$1)\quad \hat{x}_{\mathrm{k|k-1}} = A_{\mathrm{k}}\hat{x}_{\mathrm{k-1k-1}} + B_{\mathrm{k}}u_{\mathrm{k}} \tag{4}$$

$$2)\quad P_{\mathrm{k|k-1}} = G_{\mathrm{k-1}}Q_{\mathrm{k-1}}G_{\mathrm{k-1}}^{\mathrm{T}} + A_{\mathrm{k-1}}P_{\mathrm{k-1|k-1}}A_{\mathrm{k-1}}^{\mathrm{T}} \tag{5}$$

and the measurement update part consists of following steps:

$$3)\quad K_{\mathrm{k}} = P_{\mathrm{k|k-1}}C_{\mathrm{k}}^{\mathrm{T}}\left(C_{\mathrm{k}}P_{\mathrm{k|k-1}}C_{\mathrm{k}}^{\mathrm{T}} + F_{\mathrm{k}}R_{\mathrm{k}}F_{\mathrm{k}}^{\mathrm{T}}\right)^{-1} \tag{6}$$

$$4)\quad \hat{x}_{\mathrm{k|k}} = \hat{x}_{\mathrm{k|k-1}} + K_{\mathrm{k}}\left(y_{\mathrm{k}} - C_{\mathrm{k}}\hat{x}_{\mathrm{k|k-1}}\right) \tag{7}$$

$$5)\quad P_{\mathrm{k|k}} = \left(I - \mathrm{K_k}C_{\mathrm{k}}\right)P_{\mathrm{k|k-1}} \tag{8}$$

where $\hat{x}$ is the estimated state vector, P is the estimating covariance matrix and K is the Kalman gain.

However, one problem of a Kalman filter is that it requires some characteristics of the noise. Q and R are the covariance matrices of process and measurement noises, respectively, and these are usually not known. To overcome this problem, inspired by the proposed method in [15], we propose a particle swarm optimization (**PSO**) and combine this with the Kalman filter. In this method, first, in offline mode using a combination of **PSO** and the Kalman filter, the optimal values of Q and R can be found. Then, these obtained values are injected into the Kalman filter for estimation in online mode.

Using this Kalman filter, state variables of the system are estimated. The system's output can also be estimated based on these state variables and using the system model as follows:

$$\hat{y}_{\mathrm{k}} = C\hat{x}_{\mathrm{k}} \tag{9}$$

Now by comparing the estimated signal $\hat{y}_k$ with the output signal $\tilde{y}_k$, which is virtual version of the measurement signal y and regenerated by the digital twin and exactly follows y, the residual signal is generated:

$$r_k = \tilde{y}_k - \hat{y}_k \tag{10}$$

In order to distinguish attacks from noises and detect the occurrence of an attack, it is necessary to use a detector. Actually, the detector helps to make the effects of attacks and noises on the signal more prominent and filter impacts of noises and prevent false alarms. Our detector uses residual signal r and generates h so as to detect attacks as follows:

$$h_k = \sup_{k-k_0<i<k} |r_i|, \quad \begin{cases} H_0: \text{ if } & h_k \leq \text{ threshold} \\ H_1: \text{ if } & h_k > \text{ threshold} \end{cases} \tag{11}$$

where the hypothesis H_0 indicates the normal operation of the system and H_1 indicates the abnormal mode of the system.

Since the Kalman filter cannot distinguish the changes caused by the attack from the changes due to the presence of the noise, a threshold for filtering the effects of noise and preventing false alarms should be considered. Based on 68- 95-99.7 rule, in a Gaussian distribution, 68.27%, 95.45%, and 99.73% of the values lie within one, two, and three standard deviations of the mean, respectively. Since the noise in this paper is assumed to be Gaussian noise with zero mean, by considering a threshold equal to 3σ, where σ, is the standard deviation of the measurement noise, 99.73% false alarms that may occur due to this noise can be filtered out.

3.2 ATTACK CLASSIFICATION

The goal of the classification is to categorize data into distinct classes. Support Vector Machine (**SVM**) is a machine learning approach used for classification, and in this approach, a model is generated based on training data and then it can be used to predict the class of new data. **SVM** supports both binary classification and multi classification. In this paper, we want to classify the state of the system as Normal, Scaling attack or Ramp attack. Therefore, a multi classification must be used. There are several methods for multi-class **SVM**, but we apply the one-against-one method which is the most efficient one based on [16]. In the one-against-one method, $k(k-1)/2$ classifiers are created: $\left(w^1\right)^T \phi(x) + b^1$, ... , $\left(w^{k(k-1)/2}\right)^T \phi(x) + b^{k(k-1)/2}$ where k is the number of classes. Each of these classifiers is trained on data from two classes. For training data from the ith and the jth classes, the following binary classification problem is solved:

$$\begin{aligned}\min_{w^{ij},b^{ij},\xi^{ij}} \quad & \frac{1}{2}\left(w^{ij}\right)^{T} w^{ij} + C\sum_{t}\xi_{t}^{ij} \\ & \left(w^{ij}\right)^{T} \phi\left(x_t\right) + b^{ij} \geq 1 - \xi_{t}^{ij}, \text{ if } y_t = i \\ & \left(w^{ij}\right)^{T} \phi\left(x_t\right) + b^{ij} \leq -1 + \xi_{t}^{ij}, \text{ if } y_t = j \\ & \xi_{t}^{ij} \geq 0\end{aligned} \tag{12}$$

where ϕ is the function that maps training data x_t to a higher dimensional space to make data more separable and C is the penalty parameter. By minimizing $\frac{1}{2}(w^{ij})^T w^{ij}$ we maximize $\frac{2}{\|w^{ij}\|}$, which is the margin between two groups of data. When data are not linear separable, there is a penalty term $C\sum_{t}\xi_{t}^{ij}$ which can reduce the number of training errors. Actually SVM searches for a balance between the regularization term $\frac{1}{2}(w^{ij})^T w^{ij}$ and the training errors. After all $k(k-1)/2$ classifiers are constructed, if sign $\left((w^{ij})^T\phi(x) + b^{ij}\right)$ says x is in the ith class, then the vote for the ith class is added by one. Otherwise, the jth is increased by one. Then, x is predicted as the class with the largest vote, and in case that two classes have identical votes we select the one with the smaller index.

In this paper, we propose using the residual signal as x_t for classification. Our motivation behind this decision is that the residual signal is the result from the comparison between the virtual version of the real output signal and the estimated output. Hence, it shows abnormal behavior of the system caused by the attacks, and based on how this abnormal behavior is for each class, we can train a model for attack classification. By applying Ramp attacks and Scaling attacks and also considering the condition where there is no attack, we can generate training data. Then, by labeling these data as class 0 for Normal condition, class 1 for when there is Scaling attack, and class 2 for when there is Ramp attack, and using one-against-one method, we train a model for classification.

4 EXPERIMENTS

We evaluate our proposed approach in Matlab Simulink [17], and in this section we describe our simulation model and our experiments.

4.1 PROCESS

In this paper, we use a ball and beam process. The ball and beam system consists of a long beam which can be tilted by an electric motor together with a ball rolling back and forth on top of the beam. This system is open-loop unstable and without a controller, it will swing to one side or the other, and the ball will fall off the end of the beam.

To stabilize the ball, a control system that measures the position of the ball and adjusts the beam accordingly must be used. Our motivation for choosing the ball and beam system is that it is an intrinsically unstable and time-critical system, and any attack on this system can make it unstable quickly. So, evaluating our proposed method on this system can prove the effectiveness of it.

We simulate the ball and beam process using the standard Lagrangian equations of motion for the ball based on [18]. In our ball and beam system, the length of the beam is 1 meter. Therefore, the allowable position for the ball is between 0 and 1, and if it goes outside this range, it will fall.

4.2 DIGITAL TWIN

Based on our prior work [10], we create a digital twin for the ball and beam process in Matlab Simulink, which follows its physical counterpart continuously. The network between the physical domain and digital part (cloud), is simulated with TrueTime [19]. We consider the network to be an Ethernet with 2.5% packet loss probability and a 40 ms network delay.

4.3 GENERATING ATTACK SIGNALS

An attack should have two main features. First, it should cause the system to go to an unsafe state, and second, it should not be easily detectable. Ramp attacks add a ramp signal to the measurement signal and causes a change of the position of the ball. If the slope of this ramp signal is high, it changes the ball's position quickly. However, such an attack will be detected easily. So, the slope should be low and change the ball's position gradually in which case it is difficult to detect.

Therefore, we chose 0.5 meters, which is in the middle of the beam, as a setpoint for the ball's position in the controller, and based on this, choosing $0 < \lambda_r \leq 0.1$ in (2) for the Ramp attack is reasonable, since it will cause the ball to fall off and it will be difficult to detect.

Scaling attacks are not so different in terms of how they gradually can change the ball's position. The only main difference between Scaling attacks with different parameters λ_s, see (1), is the consequence. If $\lambda_s \geq 15$, it will cause the ball to fall. However, choosing λ_s out of this range, only causes the ball's position to change on the beam, but it does not cause it to fall and damage the system. So $\lambda_s \geq 15$ is a reasonable range for this type of attacks.

4.4 EVALUATION OF ATTACK DETECTION METHOD

The performance of the detection algorithm is evaluated by measuring the time it takes to detect an attack and comparing it with the time it takes the attack to drive the system to an unsafe state. Here, for the ball and beam system, we compare the time it takes our detection method to detect an attack with the time that the attack takes to cause the ball to fall off.

For this evaluation, we apply Ramp attacks and Scaling attacks with different parameters chosen from the selected range to the measurement signal in the physical domain y. Then, we record the time it takes to detect the attack and the time it takes for the attack to cause the ball to fall, which is the time it takes for the position of the ball to increase more than 1 meter or decrease less than 0 meters, and then we compare these two time.

4.5 EVALUATION OF ATTACK CLASSIFICATION METHOD

To classify attacks and diagnose the type of an attack, as it said before, we use residual signal r as training data for obtaining a model using SVM, and this model will classify new residual data as Normal, Scaling attack and Ramp attack.

To generate the training data, first, we run the simulation in normal conditions when there are no attacks several times for 60 s and record the residual signal. Using this signal we create a vector containing residual data related to a normal condition that we label as class 0.

In the next step, we apply Scaling attacks with different λ_s to the measurement signal y for 60 s, and for each λ_s, we record the residual signal. In this way, we create a vector containing residual data related to Scaling attacks that are labelled as class 1.

The reason for applying attacks for 60 s is that we want to diagnose the type of attack as soon as it occurs. So, we need to cover different data related to the beginning of the attack, and therefore 60 s is well enough for this purpose.

In the third step, we apply Ramp attacks with different λ_r to the measurement signal y for 60 s and for each λ_r, we record the residual signal. In this way, we create a vector containing residual data related to Ramp attacks that are labelled as class 2. By using this training data with the SVM algorithm, we obtain a model that should be able to determine the class of new data.

The next step is to test this model using testing data. To generate testing data for normal conditions, we run the simulation without applying any attacks for 50 s and record the residual signal as testing data. Testing data for Scaling attacks are generated by applying Scaling attacks at time 30 s for 20 s with different λ_s that are chosen from the range $\lambda_s \geq 15$. Testing data for Ramp attacks are generated by applying Ramp attacks at time 30 s for 20 s with different λ_r that are selected from the range $0 < \lambda_r \leq 0.1$.

To evaluate the performance of the classification algorithm, the class (label) of the testing data using the obtained SVM model is determined. The accuracy is then calculated as follows:

$$\text{Accuracy} = \frac{\text{The number of correctly predicted data}}{\text{Total number of testing data}} \times 100\% \tag{13}$$

For each of λ_s and λ_r value and normal condition, the generation and evaluation of testing data is repeated 15 times and finally, the average accuracy for each of λ_s and λ_r value and normal condition are calculated. Since the variability of accuracy for each of these conditions is really low and we have a quite narrow confidence interval, 15-time repetition is well enough.

5 RESULTS

In this section, the results of the evaluation of attack detection and classification algorithms are presented.

Figure 2 shows the signal h and the ball's position in the presence of Ramp attack with $\lambda_r = 0.04$ which is started at 30 s. In this attack, the attacker gradually changes the ball's position and as it can be seen in Figure 2, the attack causes the ball to fall off at 42.52 s. However, our attack detection algorithm can detect this attack at 31.29 s, which is well before the ball falls off.

Figure 3 shows the results for a Ramp attack which is started at 30 s with different values of λ_r. All chosen λ_r are small enough to make the attack difficult to detect. For each λ_r, the blue line shows the time it takes for the ball to fall off, and the red line shows the time it takes for our detection algorithm to detect the attack. As can be seen in the figure, all attacks can be detected before the attack can drive the system to an unsafe state and cause the ball to fall.

Figure 4 shows a similar evaluation for the Scaling attack with different λ_s. As can be seen in the figure, although these attacks can quickly affect the system and move the ball off the beam, our proposed detection technique can detect them in a timely manner and before the ball falls off.

To evaluate the classification method, as is said in the previous section, we apply a Ramp attack at time 30 s for 20 s. So, before 30 s, there are no attacks and the condition should be classified as Normal. After 30 s, the condition should be classified as a Ramp attack. By calculating the accuracy, the algorithm can be evaluated.

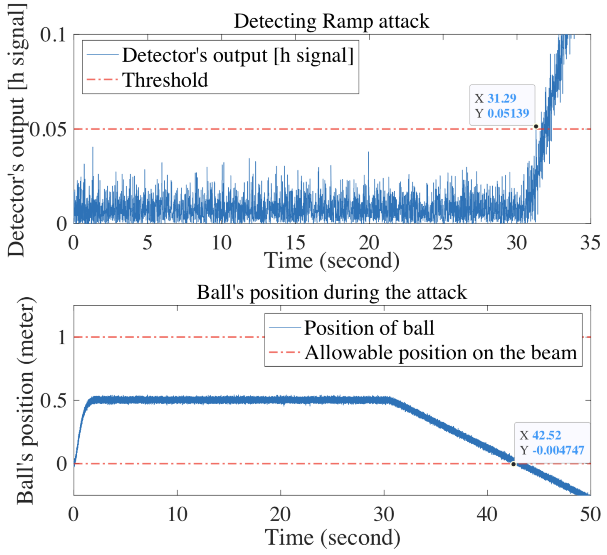


Figure 2: Detecting Ramp attack with $\lambda_r = 0.04$.

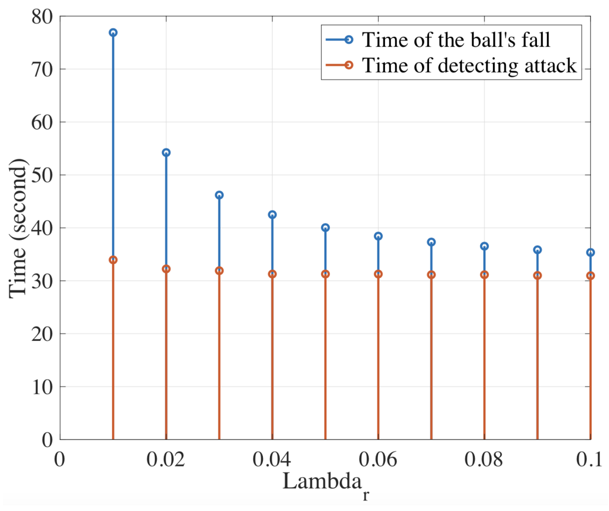


Figure 3: Detecting Ramp attacks.

Table I shows the average accuracy for the Ramp attack with different λ_r. As can be seen, the accuracy for $\lambda_r = 0.03, 0.04, 0.05$ decreases. The reason for this result is that, in this interval, there is an overlap with other classes. However, for other λ_r the accuracy is more than 91%. Also the total average

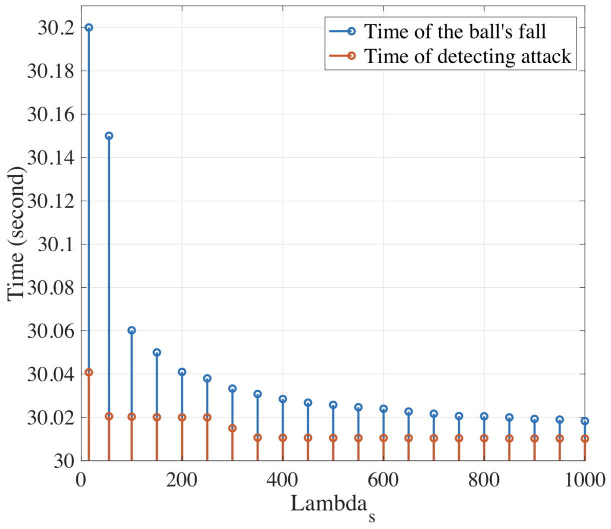


Figure 4: Detecting Scaling attacks.

accuracy of all cases is 91.27%, which is a high accuracy that shows that our classification method can recognize a Ramp attack.

Similar to the Ramp attack, we apply a Scaling attack to the system at time 30 s for 20 s and calculate the accuracy. Table II shows the average accuracy for the Scaling attack with different λ_s. The accuracy is more than 98% for all cases, and the total average accuracy is 98.96%, which proves that our classification method can recognize a Scaling attack.

Finally, we evaluate the accuracy when there are no attacks. In this case, all data should be classified as Normal. The average accuracy for this condition equals 99.94% that shows that our classification method can recognize Normal conditions excellently.

6 CONCLUSIONS

Industrial control systems are increasingly being connected to communications networks, which make them more vulnerable to cyber attacks. Regarding this issue, in this paper, we propose a digital twin based intrusion detection technique. By deploying the intrusion detection system in the digital domain, more advanced methods that require more computing resources can be developed. Therefore, we have developed and evaluated an intrusion detection mechanism for the digital twin, which can both detect attacks and also classify the type of attack. The intrusion detection mechanism uses a

Table I: Accuracy of Ramp Attack Classification

Parameter λ_r	**Accuracy**
0.01	91.42%
0.02	94.77%
0.03	83.97%
0.04	87.48%
0.05	89.88%
0.06	91.37%
0.07	92.44%
0.08	93.10%
0.09	93.85%
0.1	94.42%

Table II: Accuracy of Scaling Attack Classification

Parameter λ_s	**Accuracy**
20	98.97%
55	99.01%
100	99.03%
200	98.94%
300	98.95%
400	98.94%
500	98.93%
600	98.96%
700	98.93%
800	99.10%
900	98.92%
1000	98.95%

combination of a Kalman filter to detect the attack, a particle swarm optimization algorithm to estimate the noise, and a support vector machine algorithm to classify the attack. Through simulation studies in Matlab Simuling, we show that our detection method is highly effective to detect an attack before the attack can drive the system to an unsafe state. Also, we show that our classification approach can provide a high accuracy when determining the types of attacks. Therefore, our proposed approach can assist industrial control systems with detecting cyber attacks before they cause damage and also classify the type of attack, which will be of great benefit when it comes to choosing the proper mitigation method for each type of attack.

Bibliography

[1] J. Cheng, W. Chen, F. Tao, and C.-L. Lin, “Industrial IoT in 5G environment towards smart manufacturing,” *Journal of Industrial Information Integration*, vol. 10, pp. 10–19, 2018.

[2] R. Langner, “Stuxnet: Dissecting a cyberwarfare weapon,” *IEEE Security & Privacy*, vol. 9, no. 3, pp. 49–51, 2011.

[3] D. Alert, “Cyber-attack against ukrainian critical infrastructure,” *Cybersecurity Infrastruct. Secur. Agency, Washington, DC, USA, Tech. Rep. ICS Alert (IR-ALERT-H-16-056-01)*, 2016.

[4] ICS-CERT, “Hatman—safety system targeted malware,” Mar. 2017. [Online]. Available: https://ics-cert.us-cert.gov/MAR-17-352-01-HatManTargeted-Malware.

[5] Kaspersky Lab ICS-CERT, “Threat landscape for industrial automation systems in h2 2017,” Mar. 2018. [Online]. Available: https://icscert.kaspersky.com/reports/2018/03/26/threat-landscape-for-industrial-automation-systems-in-h2-2017/.

[6] N. S. Malik, R. Collins, and M. Vamburkar, “Cyber-attack, pings data systems of at least four gas networks,” Apr. 2018. [Online]. Available: https://www.bloomberg.com/news/articles/2018-04-03/day-after-cyberatta ck-a-third-gas-pipeline-data-system-shuts.

[7] M. Farsi, A. Daneshkhah, A. Hosseinian-Far, and H. Jahankhani, *Digital Twin Technologies and Smart Cities*. Springer, 2020.

[8] M. Eckhart and A. Ekelhart, “Towards security-aware virtual environments for digital twins,” in *Proceedings of the 4th ACM workshop on cyber-physical system security*, 2018, pp. 61–72.

[9] ——, "A specification-based state replication approach for digital twins," in *Proceedings of the 2018 Workshop on Cyber-Physical Systems Security and Privacy*, 2018, pp. 36–47.

[10] F. Akbarian, E. Fitzgerald, and M. Kihl, "Synchronization in digital twins for industrial control systems," arXiv e-prints, p. arXiv: 2006.03447, June 2020.

[11] Y.-L. Huang, A. A. Cárdenas, S. Amin, Z.-S. Lin, H.-Y. Tsai, and S. Sastry, "Understanding the physical and economic consequences of attacks on control systems," *International Journal of Critical Infrastructure Protection*, vol. 2, no. 3, pp. 73–83, 2009.

[12] S. Sridhar and M. Govindarasu, "Model-based attack detection and mitigation for automatic generation control," *IEEE Transactions on Smart Grid*, vol. 5, no. 2, pp. 580–591, 2014.

[13] F. Akbarian, A. Ramezani, M.-T. Hamidi-Beheshti, and V. Haghighat, "Intrusion detection on critical smart grid infrastructure," in *2018 Smart Grid Conference (SGC)*. IEEE, 2018, pp. 1–6.

[14] D. Simon, *Optimal state estimation: Kalman, H infinity, and nonlinear approaches*. John Wiley & Sons, 2006.

[15] Y. Laamari, K. Chafaa, and B. Athamena, "Particle swarm optimization of an extended kalman filter for speed and rotor flux estimation of an induction motor drive," *Electrical Engineering*, vol. 97, no. 2, pp. 129–138, 2015.

[16] C.-W. Hsu and C.-J. Lin, "A comparison of methods for multiclass support vector machines," *IEEE transactions on Neural Networks*, vol. 13, no. 2, pp. 415–425, 2002.

[17] MATLAB, *9.7.0.1190202 (R2019b)*. Natick, Massachusetts: The MathWorks Inc., 2018.

[18] Ball and Beam: Simulink Modeling. [Online]. Available: http://ctms.engin.umich.edu/CTMS/index.php?example=BallBeam&section\=SimulinkModeling.

[19] A. Cervin, D. Henriksson, B. Lincoln, J. Eker, and K.-E. Arzen, "How does control timing affect performance? analysis and simulation of timing using jitterbug and TrueTime," *IEEE control systems magazine*, vol. 23, no. 3, pp. 16–30, 2003.

Paper II

Paper II

Reproduced, with permission, from

FATEMEH AKBARIAN, WILLIAM TÄRNEBERG, EMMA FITZGERALD, MARIA KIHL, "A Security Framework in Digital Twins for Cloud-based Industrial Control Systems: Intrusion Detection and Mitigation," *The 26th IEEE International Conference on Emerging Technologies and Factory Automation (ETFA)*, Västerås, Sweden, Sep. 2021.

II

A Security Framework in Digital Twins for Cloud-based Industrial Control Systems: Intrusion Detection and Mitigation

With the help of modern technologies and advances in communication systems, the functionality of Industrial Control System (**ICS**) has been enhanced leading toward to have more efficient and smarter **ICS**. However, this makes these systems more and more connected and part of a networked system. This can provide an entry point for attackers to infiltrate the system and cause damage with potentially catastrophic consequences. Therefore, in this paper, we propose a digital twin-based security framework for **ICS** that consists of two parts: attack detection and attack mitigation. In this framework we deploy an intrusion detection system in digital domain that can detect attacks in a timely manner. Then, using our mitigation method, we keep the system stable with acceptable performance during the attack. Additionally, we implement our framework on a real testbed and evaluate its capability by subjecting it to a set of attacks.

1 INTRODUCTION

Control systems form a significant part of industrial systems and they play an important role in monitoring and controlling, for example, physical and chemical processes. With the advent of new technology and the expansion of telecommunication networks, Industrial Control Systems (**ICS**) are becoming more connected and part of networked systems. Hence, components of **ICS** can be distributed over a large area, and communicate with each other via wired or wireless links. This can enable remote monitoring and control of the system. Along with their numerous benefits, there are growing concerns about the safety and security of **ICS**, since they can provide a critical platform through which attackers may be able to infiltrate the system. In recent years, there have been a number of attacks that targeted **ICS** such as the Stuxnet attacks on Iran's nuclear installation in 2010 [1], the BlackEnergy malware attack on the Ukrainian power grid in 2015 [2], the HatMan malware attack on critical infrastructure using Schneider Electrics Safety Instrumented Systems in 2017 [3], the malware attack on 40 percent of all **ICS** in energy organizations protected by Kaspersky Lab solutions in 2017 [4], and the Cyber attacks on three U.S. natural gas pipeline companies in 2018 [5]. This demonstrates that **ICS** infrastructure is prone to cyber attacks and it highlights the necessity for appropriate security measures to protect them.

Consequently, some studies have been done in this respect. For example, some methods for anomaly detection in power grid system as networked control systems have been proposed in [6], [7], [8]. The authors in [9] have introduced different attack scenarios for networked control systems and evaluated the effects of these attacks on a real process. Besides knowledge-based attack detection techniques, some data-driven anomaly detection using machine learning approaches have also been proposed in [10], [11], [12].

Developments in cloud computing are paving the way for control of industrial control systems over the cloud. The cloud provides seemingly endless computing and storage resources that can be used to execute more advanced control strategies, allowing the controller to evaluate complex problems that are too computationally demanding to perform locally. However, connecting these systems to the cloud through the network will provide an access point for attackers to intrude into the system. Hence, in this paper, we propose a digital twin-based security framework to detect and mitigate attacks on **ICS** in a timely manner.

Digital twin is a rather new concept associated with **ICS**, and it opens up new possibilities in terms of monitoring, simulating, optimizing and predicting the state of the systems. With digital twins, we have virtual replicas of physical systems so that they precisely mirror the internal behavior of the physical systems [13]. Digital twin based intrusion detection and mitigation has some advantages. As the digital twin is deployed in the cloud, an intrusion detection system in the digital domain (cloud) can include methods that require much more computing resources than if deployed in the real system. Also, our intrusion detection is implemented close to the controller when we control the system over the cloud and the controller is deployed in the cloud. Hence, the intrusion detection has access to the signals that are sent to the cloud for the controller and can evaluate whether they are healthy or manipulated.

A few papers have suggested intrusion detection as a use case for digital twins. The authors in [14] defined two rules, namely, safety and security rules, that specific digital twins must adhere to. The safety rule defines a threshold for a variable, for example, the maximum velocity of a motor that the PLC controls, and the security rule specifies a consistency check between a tag of the PLC and a tag of the machine interface (HMI). For instance, when the velocity of a motor is set using an HMI and sent to a PLC, the value of the velocity on these two devices should match. So, during the operation of the systems, the digital twins are checked continuously for any rule violations. However, this research missed the synchronization between digital twins and the real systems, and regarding this issue, the authors in [15] proposed a passive state replication to create synchronization between physical systems and digital twins, which is a fundamental requirement for realizing intrusion detection using digital twins. In this method, only the inputs of the physical system that constitutes a stimulus should be fed to the digital twin. For example, a set-point that a user chooses for the system through the HMI is a kind of data that should be replicated in the digital twin. The authors have also proposed a behavior-specification-based intrusion detection to determine whether the system's behavior during runtime diverges from the predefined correct behavior due to an intrusion. However, creating the specification of the system's correct behavior typically requires processing effort, whereas in this paper the authors sidestep this issue by making the assumption that the specification of the system is readily available.

In our prior work [16], we proposed an architecture to implement a digital twin and we equipped this architecture with a novel intrusion detection algorithm that unlike [15] does not need a specification of the system's correct behaviour. Then, we evaluated the capability of our proposed anomaly detection algorithm through simulation studies. In this paper, we propose a digital twin-based security framework for industrial control systems. In this security framework, we develop a novel algorithm that is deployed in the digital twin for timely detection of attacks. Also, we propose an approach to mitigate the attack after detection. Finally, as a proof of concept,

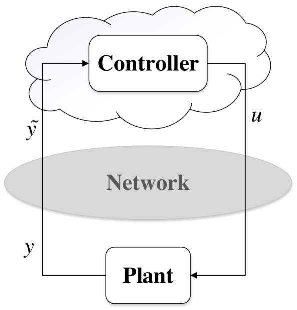


Figure 1: Cloud Control Systems overview.

we implement our proposed security framework on a real testbed and we demonstrate its effectiveness for timely detection and mitigation of attacks.

2 CLOUD CONTROL SYSTEMS

Our targeted system in this paper is Cloud Control System (**CCS**) that is illustrated in Figure 1. The system consists of a plant controlled by a controller implemented in the cloud. The plant is modeled as follows:

$$\begin{aligned} x_{\mathrm{k}+1} &= Ax_{\mathrm{k}} + Bu_{\mathrm{k}} + Gw_{\mathrm{k}} & w &\rightarrow N(0, Q) \\ y_{\mathrm{k}} &= Cx_{\mathrm{k}} + Fv_{\mathrm{k}} & v &\rightarrow N(0, R) \end{aligned} \tag{1}$$

In this model x is the state vector, y is the measurement signal, u is the control signal, w is process noise, v is measurement noise and subscript k shows the time instance. Here, we consider process noise and measurement noise to be white noise with covariances Q and R respectively. Also, A, B, C, G, and F are coefficient matrices.

The measurement signal y is measured by sensors and sent to the controller in the cloud. Then, the controller, using the measurement signal, generates control signal u and sends it back to the plant. Since these signals are sent through the telecommunication link between the plant and the controller, there is the possibility of attacks occurring in which the signals are manipulated by the attacker. In this paper, we assume the attacker tries to manipulate measurement signal y and we will investigate occurring attack on control signal u in our future work.

We consider a "man in the middle" attack in this work, and in this type of attack, the attacker can secretly listen to the values transmitted between processes and controllers in the lower layer and has the ability to manipulate or corrupt them. So, the attacker adds an attack vector $a = [a_1 a_2 ... a_N]^T$ to the measurement signal that has nonzero entries for measurements under attack and zero values for all other measurements. Consequently, the controller will receive $\tilde{y}$ instead of y that is defined as follows:

$$\tilde{y} = y + a \tag{2}$$

where $\tilde{y}$ is the manipulated measurement signal that is received by the controller and makes it generate the wrong control signal such that it can cause a critical condition for the plant.

3 PROPOSED SECURITY FRAMEWORK

The different actions to protect **ICS** from cyber attacks can be classified as prevention, detection, and mitigation. Prevention aims at decreasing the likelihood of attacks by reducing the vulnerability of the system components, for instance by encrypting the communication channels, using firewalls, and security protocols. On the other hand, detection is an approach in which the system is continuously monitored for anomalies caused by adversary actions and once an attack is detected, mitigation actions try to reduce impacts of attack on the system [17]. Our aim in this paper is designing methods related to detection and mitigation actions.

In this section, we propose a framework to ensure the stability of the plant and having acceptable performance under attacks. Actually, using this framework, we detect attacks in a timely manner and then mitigate them to diminish the effects of the attack on the plant.

Figure 2 demonstrates an overview of our proposed security framework. As shown in this Figure, we have a plant that is controlled over the cloud using the main controller. We assume an attacker intrude into the system through the network and manipulate measurement signal y sent to the main controller. In order to detect this attack, we design an intrusion detection system in the cloud that consists of a residual generator and a decision system. Decision system receives the residual signal that is generated using $\tilde{y}$ and u, and it triggers an alarm signal if it detects an attack. After detecting the attack, the alarm signal is sent to the switch and causes it to use signal $u^{'}$ generated by the ancillary controller that we have employed in physical domain, close to the plant, instead of signal u generated by the main controller.

In this paper, we assume control signal u and alarm signal are protected using some prevention actions and they are not accessible by the attacker, but the attacker has access to measurement signals y. Protecting all data including measurement signals can be costly to implement especially in large-scale control systems that we have a lot of sensors and we may need update the old equipment.

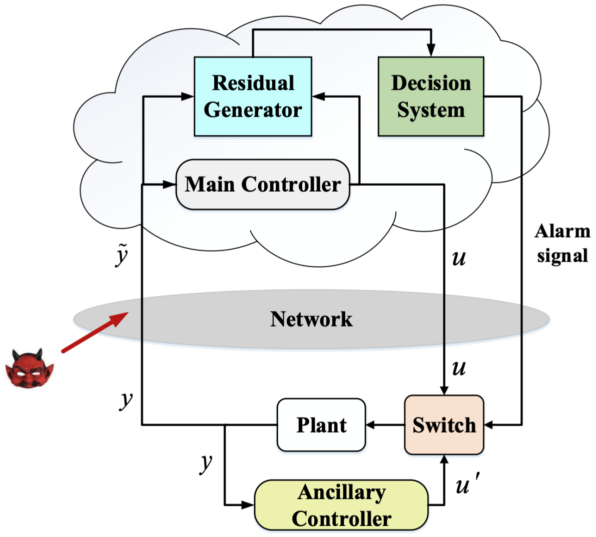


Figure 2: Security framework overview.

In the rest of this section, we explain different parts of our proposed security framework in detail.

3.1 ATTACK DETECTION

Attack detection part of our security framework consists of two parts: residual generator and decision system.

A. Residual Generator

In order to detect attacks, we try to generate a residual signal that in the absence of any attack has a nominal value close to zero and may deviate a little bit from zero only due to modelling uncertainties and noise. However, if an attack occurs, the residual deviates from zero with a magnitude such that the new condition can be distinguished from the attack-free mode.

To generate such a residual signal, we can use an observer to estimate the output of the system $\hat{y}$ and then compare it with the real output of the system y. For this purpose, we choose a Kalman filter, which is an optimal observer for stochastic processes and can estimate the behaviour of the system by using input u and output y signals of the plant. A Kalman filter for the system in (1) will be designed using the following recursive algorithm, which consists of two parts: time update and measurement update [18]. The time update part consists of the following steps:

$$1)\quad \hat{x}_{\mathrm{k|k-1}} = A_{\mathrm{k}}\hat{x}_{\mathrm{k-1k-1}} + B_{\mathrm{k}}u_{\mathrm{k}} \tag{3}$$

$$2)\quad P_{\mathrm{k|k-1}} = G_{\mathrm{k-1}}Q_{\mathrm{k-1}}G_{\mathrm{k-1}}^{\mathrm{T}} + A_{\mathrm{k-1}}P_{\mathrm{k-1|k-1}}A_{\mathrm{k-1}}^{\mathrm{T}} \tag{4}$$

and the measurement update part consists of the following steps:

$$3)\quad K_{\mathrm{k}} = P_{\mathrm{k|k-1}}C_{\mathrm{k}}^{\mathrm{T}}\left(C_{\mathrm{k}}P_{\mathrm{k|k-1}}C_{\mathrm{k}}^{\mathrm{T}} + F_{\mathrm{k}}R_{\mathrm{k}}F_{\mathrm{k}}^{\mathrm{T}}\right)^{-1} \tag{5}$$

$$4)\quad \hat{x}_{\mathrm{k|k}} = \hat{x}_{\mathrm{k|k-1}} + K_{\mathrm{k}}\left(y_{\mathrm{k}} - C_{\mathrm{k}}\hat{x}_{\mathrm{k|k-1}}\right) \tag{6}$$

$$5)\quad P_{\mathrm{k|k}} = \left(I - \mathrm{K}_{\mathrm{k}}C_{\mathrm{k}}\right)P_{\mathrm{k|k-1}} \tag{7}$$

where P is the estimating covariance matrix, K is the Kalman gain and $\hat{x}$ is the estimated state vector. The system's output can also be estimated based on these estimated states and using the system model as follows:

$$\hat{y}_{\mathrm{k}} = C\hat{x}_{\mathrm{k}} \tag{8}$$

After a transient time, the signal estimated by the kalman filter converges to the real output signal, but if an attack occurs, it will cause a difference between these two signals. Although this difference can be temporary and in steady state, the estimated signal converges to the signal manipulated by the attacker, this temporary difference can be used as a sign for an abnormal condition. Hence, by comparing this estimated signal $\hat{y}$ with the real signal y we can generate a residual signal as follows:

$$r_k = y_k - \hat{y}_k \tag{9}$$

Hence, we have as many residual signals as measurement signals.

B. Decision System

Model uncertainties, disturbances and measurement noise are not exactly annihilated in the residual, and they could drive the residual away from zero. Hence, we need a decision function for residual evaluation that will determine whether an attack is present. The decision function consists of two elements: a test function and a threshold function.

- Test Function: a test function $\varphi(r_k)$ provides a measure of the residual's deviation from zero. Some common test functions have been introduced in [19] and based on that, we consider the absolute value as a test function, is defined as follows:

$$\varphi(r_k) = |r_k| \tag{10}$$

  In Section 6 we show this simple test function works well for our objective.
- Threshold Function: a threshold function $\Phi(k)$ that we need for evaluation of the test function $\varphi(k)$ should have the following properties:

$$\begin{cases} H_0 : \varphi(r_k) \leq \Phi(k) \\ H_1 : \varphi(r_k) > \Phi(k) \end{cases} \tag{11}$$

  where the hypothesis H_0 indicates normal operation of the system and H_1 indicates the abnormal mode of the system that triggers an alarm signal.
  A set of healthy data of residual signals, during which no attacks occur, is used to calculate appropriate thresholds for each residual signal. For this purpose, we determine the threshold based on the maximum value of the residual signal during healthy conditions such that it satisfies the conditions in (11). Then, a given sample of measurement signals is classified as an anomaly if one of the residual signals exceeds the corresponding threshold.

3.2 ATTACK MITIGATION

Once the attack has been detected, attack mitigation should guarantee that the plant remains stable and has acceptable performance under abnormal states. As mentioned in Section 2, in our targeted system we have the controller in the cloud and the communication link between the plant and the cloud enables attackers to intrude into the system. Regarding control over the cloud, [20] has proposed to use basic local control as an ancillary controller. This ancillary controller is switched in to recover when networked control fails due to the loss of real-time execution and network degradation. Hence, we can use the same idea for mitigating attacks, and employ a local controller that we can switch to upon detecting an attack. Thus, we design a controller and place it in the physical domain close to the plant such that there is no public network between the controller and the plant and consequently it is secure and there is no possibility of intrusion into the system by attackers. Measurement signals are sent from the sensors to both controllers and the both generate control signals but the priority is to use the control signal generated by the controller in the cloud, since it is a more advanced controller. Once an attack has been detected, as shown in Figure 2, an alarm signal is triggered and sent by the decision system to the switch, causing it to use the control signal generated by local controller.

4 TESTBED DESCRIPTION

As a proof of concept for our proposed security framework, we implement it on a real test-bed based on [21].

4.1 PLANT

We use a ball and beam process as our plant. The ball and beam system consists of a long beam which can be tilted by an electric motor together with a ball rolling back and forth on top of the beam. This system is open-loop unstable and without a controller, it will swing to one side or the other, and the ball will fall off the end of the beam. Our motivation for choosing the ball and beam system is that it is an intrinsically unstable and time-critical system such that any attack on this system can make it unstable quickly. So, evaluating our proposed security framework on this system can prove its effectiveness.

For designing controllers and Kalman filter we need the discrete model of the plant. The ball and beam process is modeled in continuous time as follows:

$$\begin{aligned} \dot{x}(t) &= \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & -7.00475 \\ 0 & 0 & 0 \end{bmatrix} x(t) + \begin{bmatrix} 0 \\ 0 \\ 0.44 \end{bmatrix} u(t) \\ y(t) &= \begin{bmatrix} 10.18182 & 0 & 0 \\ 0 & 0 & 12.73239 \end{bmatrix} x(t) \end{aligned} \tag{12}$$

and we discretize it with sampling time 0.05 s.

4.2 KUBERNETES CLUSTER

The testbed has been equipped with a seven-node Kubernetes cluster as the edge cloud. Kubernetes (K8S) is a portable, extensible, open-source platform for managing containerized workloads and services, that facilitates both declarative configuration and automation [22]. The cluster has been equipped with an nginx ingress [23] and prometheus operator [24]. The nginx ingress is exposed using the K8S NodePort paradigm. We use this K8S cluster to implement our main controller and attack detection algorithm.

4.3 MAIN CONTROLLER

To stabilize the ball, we need a feedback controller that uses measurement signals to adjust the beam accordingly. A Model Predictive Control (**MPC**)-based controller is designed based on [20] as the main controller in Figure 2. **MPC** is an advanced method of process control that is used to control a process while satisfying a set of constraints. **MPC** relies on a dynamic model of the process and uses this model to forecast system behaviour. Hence, the main advantage of this controller is the fact that it allows the current time-slot to be optimized, while taking future time-slots into account. The current control action is obtained by solving, at each sampling instant k, a finite horizon (N) open-loop optimal control problem, using the current state of the plant as the initial state as follows:

$$\begin{aligned}
&\underset{\mathbf{u}}{\text{minimize}} J = \sum_{i=k}^{k+N-1} x_i^T Q x_i + u_i^T R u_i + x_{k+N}^T P x_{k+N} \\
&\text{subjectto} \quad x_{i+1} = A x_i + B u_i \\
&\qquad G \begin{bmatrix} x_i \\ u_i \end{bmatrix} \leq g, \quad H \begin{bmatrix} x_i \\ u_i \end{bmatrix} = h \quad , x_{n+k} \in \mathcal{T}
\end{aligned} \tag{13}$$

where Q, R and P are cost matrixes, A and B define the model of the system, x is the state vector, u is the control signal, and the constraints of the system are defined by the matrices and vectors G, g, H and h. This optimization yields an optimal control sequence $u(k), u(k+1), ..., u(k+N-1)$ and the first control in this sequence $u(k)$ is applied to the plant.

Since **MPC** is implemented in the cloud, in this paper, in order to compensate the possible network delay between the plant and the controller in the K8S cluster, we apply the future control sequence $u(k+m)$ instead of the first control value in this sequence $u(k)$. Actually, at each sampling instant k, we measure the Round-Trip Time (**RTT**) the time it takes to send the measurement signal and receive the control signal, and then by dividing this time by the sampling time we calculate m, and we decide which step of the control sequence $u(k+m)$ to use instead of the first step $u(k)$.

In this test-bed, **MPC** is implemented using Python and container technology and is deployed to the K8S cluster as a pod.

4.4 ANCILLARY CONTROLLER

The test-bed has been equipped with a local controller that is implemented as a Linear–quadratic regulator (**LQR**). The control signal of the **LQR** is generated as follows:

$$u_k = -K x_k \tag{14}$$

where K is the gain vector and x is the state vector. **LQR** is a simpler controller than **MPC** and it requires little computational capacity, allowing it to be implemented in the physical domain. This controller is implemented close to the plant and there is no network between it and the plant which means that it is secure and the attacker does not have access to the measurement signals that are sent to this controller. This controller receives the measurement signal and works in parallel with the **MPC** controller, but we use its control signal to keep the system stable only when an anomaly is detected at the **MPC** controller.

5 EXPERIMENTS

We evaluate our proposed security framework in the test-bed that was explained in Section 4, and in this section we describe our experiments.

5.1 GENERATING ATTACK SIGNALS

An attack should have two main features. First, it should cause the system to go to an unsafe state, and second, it should not be easily detectable. In this paper, we consider ramp attacks based on [6] and [8] and we assume they occur on the measured position signal that is sent to the **MPC** controller in the K8S cluster, and cause a change in the position signal.

In this type of attack, from the beginning of the attack, a value $\lambda_r.t$ is added to the actual signal, which depending on the magnitude of the slope of the ramp signal λ_r, increases or decreases the value of the signal:

$$\tilde{y}(t) = \begin{cases} y(t) & \text{for } t \notin \tau_a \\ y(t) + \lambda_r.t & \text{for } t \in \tau_a \end{cases} \tag{15}$$

If λ_r is high, it changes the ball's position quickly. However, such an attack will be detected easily. So, the slope should be low and change the ball's position gradually, in which case it is difficult to detect. The length of the beam equals 1.1 meters and the allowed range for the position of the ball is [-0.55 m, 0.55 m]. We chose -0.3 meters as a set-point for the ball's position in the controller. Based on this, choosing $0 < \lambda_r \leq 0.05$ in (15) for the ramp attack is reasonable, since it will cause the ball to fall off but it will be difficult to detect. So, if we can detect these attacks, we will be able to detect ramp attacks with larger λ_r as well.

5.2 EVALUATION OF SECURITY FRAMEWORK

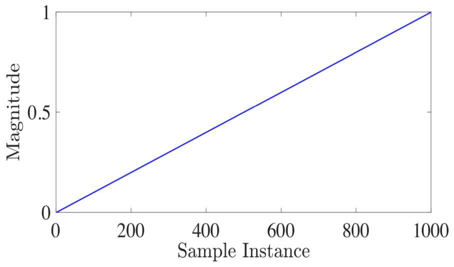
For evaluation, we performed experiments by applying ramp attacks with each $\lambda_r \in S$ that $S = \{0.001, 0.0013, 0.0015, 0.0017, 0.01, 0.013, 0.015, 0.017, 0.02,$

$0.023, 0.025, 0.027, 0.03, 0.033, 0.035, 0.037, 0.04, 0.043, 0.045, 0.047, 0.05\}$. Measurement signal y that is sent to the **MPC** controller in the cloud contain measured position of the ball, and measured angel of the beam, and in our experiment we assume the attack occurs on the measured position signal. We run each experiment for 15000 samples and we applied the ramp attack at sample k=14500 on the the measured position signal. To evaluate our attack detection method, we measured how much time it took to detect the attack and trigger the alarm signal. Also, to evaluate mitigation part of our security framework, we recorded position of the ball, and measured the maximum deviation of the ball from its set-point due to the attack. We measured the maximum deviation to know whether we can mitigate the attack and turn the ball back to its set-point or not, and also to know before turning the ball back how far it could get away from its set-point.

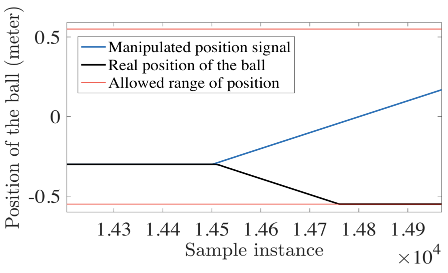
6 RESULTS AND DISCUSSION

In this section, the results of the evaluation of our security framework are presented. We start with ramp attack with $\lambda_r = 0.001$ that is the slowest and the most difficult one to detect and go into detail about it. Then results are given for a series of attacks varying the parameters.

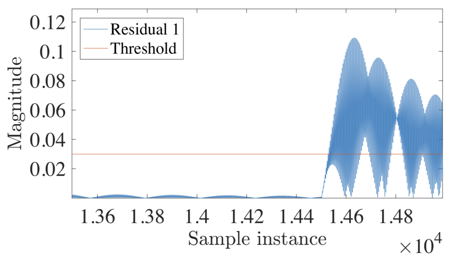
Figure 3 shows the results for a ramp attack with $\lambda_r = 0.001$. Figure 3(a) shows the attack signal: a ramp signal with slope 0.001. We assume the attacker starts to add this signal to the position signal that is sent to the cloud at the 14500th sample, and tries to increase the value of the ball's position with very low speed such that the attack cannot be detected easily. The blue curve in Figure 3(b) shows the manipulated position signal that is received by the MPC controller and deceives it into generating the wrong control signal. The black curve in Figure 3(b) shows the real position signal in the presence of this attack and as can be seen this attack drives the system to an unsafe state and causes the ball to fall off the end of the beam at the 14759th sample. Figure 3(c) and (d) show the residual signals generated using our attack detection method in the presence of this ramp attack, and the red lines in these two figures indicate the threshold. As can be seen, residual 1 at the 14531th sample and residual 2 at the 14503th sample exceed their thresholds. Residual 2 can detect the attack faster and triggers the alarm signal at the 14503th sample, as shown in Figure 3(e). After triggering the alarm, in order to mitigate the attack and recover, the system switches to the local controller. Figure 3(f) demonstrates the effectiveness of our proposed security framework. As can be seen, the set-point for the position of the ball on the beam is -0.3 m. At the 14500th sample, the attack starts to cause the ball to deviate from its set-point in order to make it off the beam. However, at the 14510th sample, when the



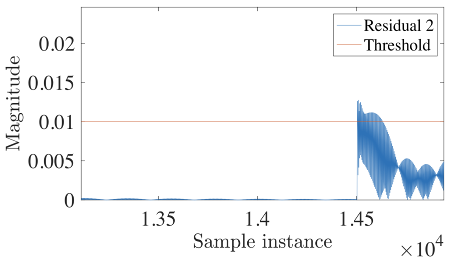
(a) Ramp Attack with slope=0.001



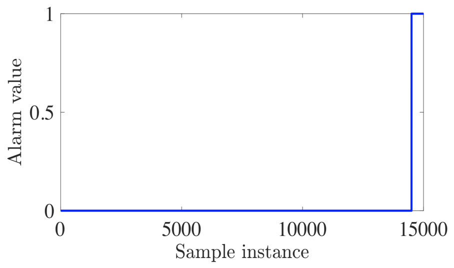
(b) Position signals



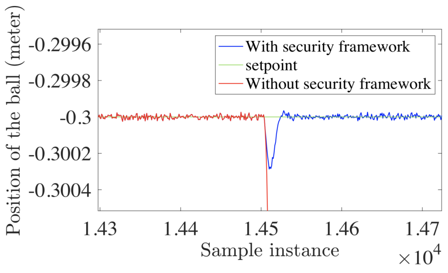
(c) Residual I signal



(d) Residual II signal



(e) Alarm signal



(f) System with security framework

Figure 3: Results for ramp attack with slope=0.001.

ball is just 0.3 mm away from its set-point, our mitigation method starts to move the ball back to the set-point. Otherwise, in the absence of our security framework, the ball continues to deviate from the set-point following the red curve in Figure 3(f), and at the end it will fall off from the end of the beam as shown by the black curve in Figure 3(b).

We also evaluated our security framework by applying ramp attacks with different λ_r, and Figure 4 shows the time it takes to detect these attacks. As we can see, an attack with $\lambda_r = 0.001$ can be detected at the third sample instance

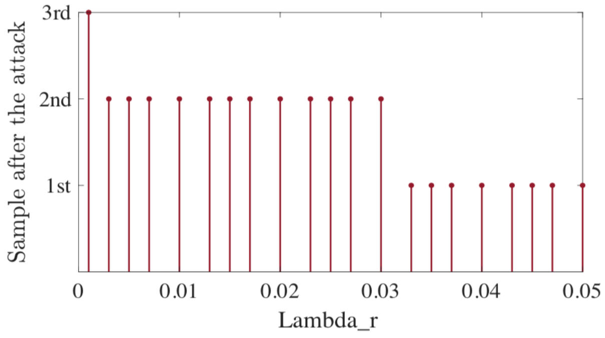


Figure 4: Detecting ramp attacks

after the start of the attack, 0.15 s after the attack starts as we use a sampling time of 0.05 s. Attacks with $0.0013 \leq \lambda_r \leq 0.03$ are detected at the second sample instance after the beginning of the attack and ramp attacks with $0.03 < \lambda_r$ are detected at the first sample instance after the beginning of the attack. The steeper the attack signal is, the greater change it makes in the position of the ball and the faster it is detected. Hence, we performed experiments, shown in Figure 4, for ramp attacks with very small slope ($0.001 \leq \lambda_r \leq 0.05$) that are very difficult to detect. We showed that these attacks can be detected very quickly using our attack detection method. Also, based on these experiments, it is clear that attacks with slope more that 0.05 are detected easily at the first step after the start of the attack.

The attacker tries to drive the ball off the beam, but in the presence of our security framework, we can detect these attacks quickly, as shown in Figure 4, and then by using our mitigation method, we turn the ball back to its set-point. Figure 5 shows the maximum deviation of the ball from its set-point caused by ramp attacks with different λ_r. When the ball reaches these maximum deviation, the mitigation part of the security framework prevents it from continuing to move in that direction and turns it back to the set-point. The maximum deviation the ball reaches depends on both the time it takes to detect the attack and the slope of the attack, which determines how fast the attack affects the system. For example, for attacks with $\lambda_r = 0.03$ and $\lambda_r = 0.04$, based on Figure 4, it takes 0.1 s and 0.05 s respectively to detect the attack. Hence, although an attack with $\lambda_r = 0.04$ is more powerful than one with $\lambda_r = 0.03$ and moves the ball faster, it is detected faster and so has a shorter time to affect the system. This means that it causes a smaller deviation than an attack with $\lambda_r = 0.03$.

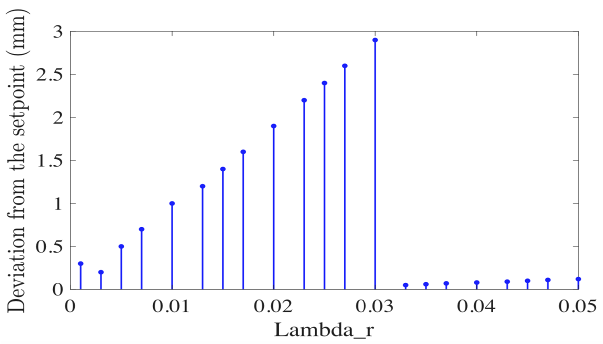


Figure 5: Maximum deviation of the ball from its set-point due to ramp attacks in the presence of the security framework

7 CONCLUSIONS

Considering the vulnerability of industrial control systems to cyber attacks especially when they are controlled over the cloud, in this paper we proposed a security framework using the concept of digital twins. In this framework, we designed a residual generator in the digital domain that can detect attacks so quickly with the help of a decision system. To mitigate the effects of attacks, we added a local controller on the factory floor close to the plant, which is switched in to recover the system when an anomaly is detected on the main controller in the cloud. We evaluated our proposed security framework using a real testbed, showing that it can detect attacks on a real system in a timely manner and keep this system stable with good performance even during attacks.

Bibliography

[1] R. Langner, “Stuxnet: Dissecting a cyberwarfare weapon,” *IEEE Security & Privacy*, vol. 9, no. 3, pp. 49–51, 2011.

[2] D. Alert, “Cyber-attack against ukrainian critical infrastructure,” *Cybersecurity Infrastruct. Secur. Agency, Washington, DC, USA, Tech. Rep. ICS Alert (IR-ALERT-H-16-056-01)*, 2016.

[3] ICS-CERT, “Hatman—safety system targeted malware,” Mar. 2017. [Online]. Available: https://ics-cert.us-cert.gov/MAR-17-352-01-HatManTargeted-Malware.

[4] Kaspersky Lab ICS-CERT, “Threat landscape for industrial automation systems in h2 2017,” Mar. 2018. [Online]. Available: https://icscert.kaspersky.com/reports/2018/03/26/threat-landscape-for-industrial-automation-systems-in-h2-2017/.

[5] N. S. Malik, R. Collins, and M. Vamburkar, “Cyber-attack, pings data systems of at least four gas networks,” Apr. 2018. [Online]. Available: https://www.bloomberg.com/news/articles/2018-04-03/day-after-cyberatta ck-a-third-gas-pipeline-data-system-shuts.

[6] F. Akbarian, A. Ramezani, M.-T. Hamidi-Beheshti, and V. Haghighat, “Advanced algorithm to detect stealthy cyber attacks on automatic generation control in smart grid,” *IET Cyber-Physical Systems: Theory & Applications*, vol. 5, no. 4, pp. 351–358, 2020.

[7] B. Li, G. Xiao, R. Lu, R. Deng, and H. Bao, “On feasibility and limitations of detecting false data injection attacks on power grid state estimation using d-facts devices,” *IEEE Transactions on Industrial Informatics*, vol. 16, no. 2, pp. 854–864, 2019.

[8] S. Sridhar and M. Govindarasu, “Model-based attack detection and mitigation for automatic generation control,” *IEEE Transactions on Smart Grid*, vol. 5, no. 2, pp. 580–591, 2014.

[9] A. Teixeira, D. Pérez, H. Sandberg, and K. H. Johansson, “Attack models and scenarios for networked control systems,” in *Proceedings of the 1st international conference on High Confidence Networked Systems*, 2012, pp. 55–64.

[10] F. Zhang, H. A. D. E. Kodituwakku, J. W. Hines, and J. Coble, “Multilayer data-driven cyber-attack detection system for industrial control systems based on network, system, and process data,” *IEEE Transactions on Industrial Informatics*, vol. 15, no. 7, pp. 4362–4369, 2019.

[11] A. Sargolzaei, K. Yazdani, A. Abbaspour, C. D. Crane III, and W. E. Dixon, “Detection and mitigation of false data injection attacks in networked control systems,” *IEEE Transactions on Industrial Informatics*, vol. 16, no. 6, pp. 4281–4292, 2019.

[12] M. Esmalifalak, L. Liu, N. Nguyen, R. Zheng, and Z. Han, “Detecting stealthy false data injection using machine learning in smart grid,” *IEEE Systems Journal*, vol. 11, no. 3, pp. 1644–1652, 2014.

[13] M. Farsi, A. Daneshkhah, A. Hosseinian-Far, and H. Jahankhani, *Digital Twin Technologies and Smart Cities*. Springer, 2020.

[14] M. Eckhart and A. Ekelhart, “Towards security-aware virtual environments for digital twins,” in *Proceedings of the 4th ACM workshop on cyber-physical system security*, 2018, pp. 61–72.

[15] ——, “A specification-based state replication approach for digital twins,” in *Proceedings of the 2018 Workshop on Cyber-Physical Systems Security and Privacy*, 2018, pp. 36–47.

[16] F. Akbarian, E. Fitzgerald, and M. Kihl, “Intrusion detection in digital twins for industrial control systems,” in *2020 International Conference on Software, Telecommunications and Computer Networks (SoftCOM)*. IEEE, 2020, pp. 1–6.

[17] A. Teixeira, K. C. Sou, H. Sandberg, and K. H. Johansson, “Secure control systems: A quantitative risk management approach,” *IEEE Control Systems Magazine*, vol. 35, no. 1, pp. 24–45, 2015.

[18] D. Simon, *Optimal state estimation: Kalman, H infinity, and nonlinear approaches*. John Wiley & Sons, 2006.

[19] M. Blanke, M. Kinnaert, J. Lunze, M. Staroswiecki, and J. Schröder, *Diagnosis and fault-tolerant control*. Springer, 2016.

[20] P. Skarin, W. Tärneberg, K.-E. Årzén, and M. Kihl, "Control-over-the-cloud: A performance study for cloud-native, critical control systems," in *2020 IEEE/ACM 13th International Conference on Utility and Cloud Computing (UCC)*. IEEE, 2020, pp. 57–66.

[21] W. Tärneberg, M. Gunnarsson, M. Kihl, and C. Gehrmann, "Demonstration: A cloud-native digital twin with adaptive cloud-based control and intrusion detection," in *The Conference on Networked Systems (NetSys 2021), in press*.

[22] Kubernetes. [Online]. Available: https://kubernetes.io

[23] Ingress-nginx. [Online]. Available: https://github.com/kubernetes/ingress-nginx

[24] Prometheus-operator. [Online]. Available: https://github.com/coreos/prometheus-operator

Paper III

Paper III

Reproduced, with permission, from

FATEMEH AKBARIAN, WILLIAM TÄRNEBERG, EMMA FITZGERALD, MARIA KIHL, "A cloud-control system equipped with intrusion detection and mitigation," *Conference on Networked Systems 2021 (NetSys 2021)*, Lübeck, Germany, Sep. 2021.

III

A cloud-control system equipped with intrusion detection and mitigation

The Cloud Control System (**CCS**) are inseparable parts of industry 4.0. The cloud, by providing storage and computing resources, allows the controllers to evaluate complex problems that are too computationally demanding to perform locally. However, connecting physical systems to the cloud through the network can provide an entry point for attackers to infiltrate the system and cause damage with potentially catastrophic consequences. Hence, in this paper, we present a demo of our proposed security framework for **CCS** and demonstrate how it can detect attacks on this system quickly and mitigate them.

1 INTRODUCTION

By adoption of new technologies, we have had several revolutions in industry and now some modern technologies like Internet of Things (**IoT**), cloud computing, etc are paving the way for smart factory that will realise industry 4.0. Industrial Control System (**ICS**) as part of industry 4.0 are becoming more efficient and smarter. However, these systems are also becoming more and more connected and part of a network systems and this communication link between different components of **ICS** can provide an access point for attackers to intrude into the system and manipulate the signals that are sent through the network. For example, the attacker by manipulating measurement signals can deceive the controller to generate a wrong control signal that can make our system unstable and lead to catastrophic consequences like what we had in recent years. In recent years, we have had several attacks in different parts of industry that demonstrate **ICS** are still prone to cyber attacks and highlight the necessity for an appropriate security measure to protect these systems.

The cloud provides seemingly endless computing and storage resources that can be used to execute more advanced control strategies in **ICS**. However, in cloud control systems (**CCS**), there is a network between the plant and the cloud that the measurement and control signals are sent through this network and this can make these systems vulnerable to cyber attacks.

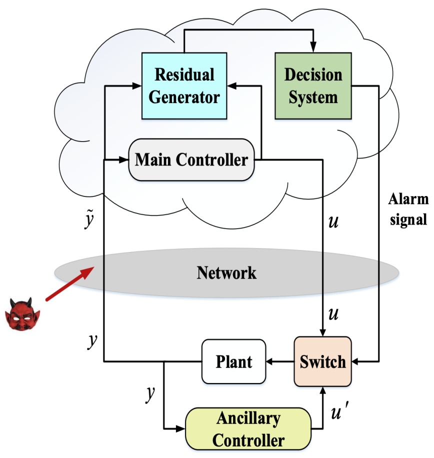
In this paper, we present a demo of our proposed security framework that is applied on **CCS** and include intrusion detection and mitigation. In this system, we have a ball and beam process as our plant, a Kubernetes (**K8S**)-cluster that hosts an intrusion detection and the main controller, and a local controller that is part of our mitigation method and implemented using Python code beside the plant.

2 SECURE CLOUD CONTROL SYSTEMS DEMO

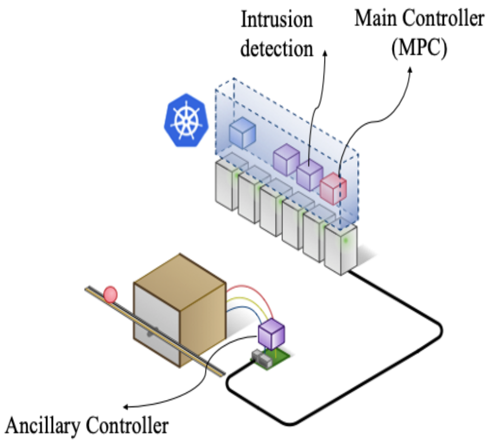
In this section, we present our demo and explain how different parts of the test-bed are implemented. Figure 1 shows an overview of our demo system. Figure 1a shows our proposed security framework, and Figure 1b shows the test-bed implementation. The system's components are detailed below.

2.1 PLANT

We use a ball and beam process as our plant. The ball and beam system consists of a long beam which can be tilted by an electric motor together with a ball rolling back and forth on top of the beam. This system is open-loop unstable and without a controller, it will swing to one side or the other, and the ball will fall off the end of the beam. Our motivation for choosing the



(a) Security framework overview.



(b) Test-bed overview.

Figure 1: An overview of the demo system.

ball and beam system is that it is an intrinsically unstable and time-critical system such that any attack on this system can make it unstable quickly. So, evaluating our proposed security framework on this system can prove its effectiveness.

2.2 KUBERNETES CLUSTER

The test-bed has been equipped with a six-node Kubernetes cluster as the edge cloud. Kubernetes (K8S) [*] is a portable, extensible, open-source platform for managing containerized workloads and services, that facilitates both declarative configuration and automation . The cluster has been equipped with an nginx ingress [†] and prometheus operator [‡] . The nginx ingress is exposed using the K8S NodePort paradigm. We use this K8S cluster to implement our main controller and intrusion detection algorithm.

2.3 MAIN CONTROLLER

To stabilize the ball, we need a feedback controller that uses measurement signals to adjust the beam accordingly. A Model Predictive Control (**MPC**)-based controller is designed based on [1] as the main controller in Figure 1.

[*] `https://kubernetes.io/`
[†] `https://github.com/kubernetes/ingress-nginx/`
[‡] `https://github.com/coreos/prometheus-operator/`

This controller for execution needs some computation resources that cloud can provide it. Thus, we deploy this controller using Python and Container technology as a pod in the Kubernetes cluster.

2.4 ATTACK AND INTRUSION DETECTION

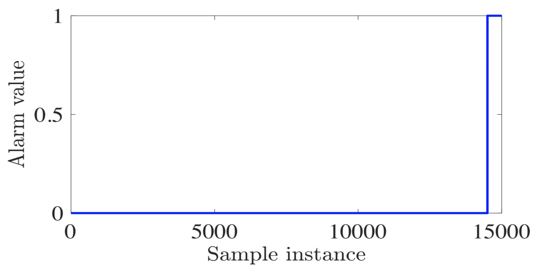
All communication between the plant and the cloud is over a LAN and uses a protocol defined in Protobuf [§] which is realized in gRPC [¶]. Measurement signals includes position of the ball, angle of the beam, and speed of the ball are sent through this network to the main controller in the cloud. The main controller using these generates the control signal and send it back to the plant. This control signal by adjusting the beam's speed controls the position of the ball on the beam. We assume the attacker tries to manipulate the measured position signal, and we implement the attack by adding a ramp signal with small slope (between 0.001 to 0.05) to the position signal using python codes.

We deploy our proposed intrusion detection in [2] using python and container technology as a pod in the Kubernetes cluster. This intrusion detection consists of two parts: residual generator and decision system. Residual generator estimates the value of the measurement signal and then by comparing it with the real one generates a residual signal. In healthy condition during which there is no attack, the residual signal is close to zero. Thus, the decision system evaluates the residual signal and by comparing it with a certain threshold decides if there is any attack in the system or not. If it detects any attack in the system, it will trigger an alarm signal.

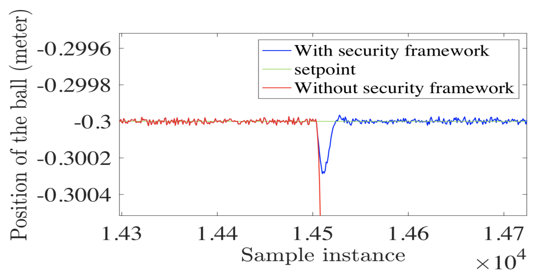
2.5 ATTACK MITIGATION METHOD AND ANCILLARY CONTROLLER

Our objective for mitigation is to keep the plant stable under the attack with an acceptable performance. As mitigation, we consider an ancillary controller that is placed close to the plant such that there is no public network between the plant and this controller so there is no possibility for the attacker to intrude into the system. Measurement signals from the plant are sent to both main and local controller, but our priority is to use the control signal generated by the main controller that is more advanced controller. Once the attack has been detected, and the alarm signal has beet triggered, we will switch to the ancillary controller. Ancillary controller is implemented as a Linear–quadratic regulator (**LQR**). We refer to [2] for a full detail of the mitigation algorithm. **LQR** is a simpler controller than **MPC** and it requires little computational capacity, allowing it to be implemented in the physical domain.

[§]`https://developers.google.com/protocol-buffers/`
[¶]`https://grpc.io/`



(a) Security framework overview.



(b) Test-bed overview.

Figure 2: Results for ramp attack with slope=0.001.

3 EXPERIMENTS

We design the attack based on section 2.4 and start applying it on the position signal at the 14500th sample. Figure 2 shows effects of this attack on the system in the presence and absence of our security framework. As it is seen in Figure 2a, Our intrusion detection can detect this attack really fast and based on this detection, our mitigation will be activated quickly. So, as the blue line in Figure 2b shows, the attack tries to deviate the ball from its setpoint, but by activating the mitigation part and switching to ancillary controller, we can move the ball back to its setpoint. Otherwise, in the absence of mitigation method, as the red line shows, the attack will move the ball to the end of the beam and finally cause the ball to fall off.

Bibliography

[1] P. Skarin, W. Tärneberg, K.-E. Årzén, and M. Kihl, "Control-over-the-cloud: A performance study for cloud-native, critical control systems," in *2020 IEEE/ACM 13th International Conference on Utility and Cloud Computing (UCC)*. IEEE, 2020, pp. 57–66.

[2] F. Akbarian, W. Tärneberg, E. Fitzgerald, and M. Kihl, "A security framework in digital twins for cloud-based industrial control systems: Intrusion detection and mitigation," in *2021 IEEE 26th International Conference on Emerging Thecnologies and Factory Automation (ETFA)*. IEEE, 2021.

Paper IV

Paper IV

Reproduced, with permission, from

FATEMEH AKBARIAN, WILLIAM TÄRNEBERG, EMMA FITZGERALD, MARIA KIHL, "Attack Resilient Cloud Control Systems," *IEEE Open Journal of Industry Applications* , submitted, 2022.

IV

Attack Resilient Cloud Control Systems

In recent years, since the cloud can provide huge advantages regarding storage and computing resources, industry has been motivated to move industrial control systems to the cloud. However, the cloud also introduces major security challenges, since moving control systems to the cloud can enable attackers to infiltrate the system and establish an attack that can lead to damages and disruptions with potentially catastrophic consequences. Therefore, some security measures are necessary to detect these attacks in a timely manner and mitigate their impact. In this paper, we propose a security framework for cloud control systems that makes them resilient against attacks. This framework includes three steps: attack detection, attack isolation, and attack mitigation. We validate our proposed framework on a real testbed and evaluate its capability by subjecting it to a set of attacks. We show that our proposed solution can detect an attack in a timely manner and keep the plant stable, with an acceptable performance during the attack.

1 INTRODUCTION

In recent years, we have had numerous advances in network technology, and these technologies have been combined with control systems to create Networked Control Systems (**NCS**). In this type of control system, the control loop is closed through the communication channel and makes it possible to monitor and adjust the plant remotely. Control systems usually have to deal with big data and this increases the communication and computational load of the network, and causes the requirements for high quality and real-time control to go beyond the traditional network control topology capability. Hence, by combining the benefits of network control and cloud computing technology, a new concept called Cloud Control System (**CCS**) has been developed that can almost solve issues of resource constrained **NCS**s. In **CCS**s, the core processing unit is shifted to a cloud server and endows the control system with massive parallel computation [1].

Although there are many benefits of combining the cloud with control systems, it leads to many security challenges. Controllers in the cloud server, and sensors in the physical domain are supposed to send packets through the communication channel, and this communication can be exposed to different types of security attacks, including passive and active attacks. In recent years, there have been a number of attacks that targeted control systems and caused damage [2–6]. This indicates the possibility of such attacks on **CCS**s and the need for appropriate security measures to protect these systems.

Computer security literature identifies three fundamental properties of information and services in IT systems, namely confidentiality, integrity, and availability, often denoted as **CIA** [7], and they can be violated by disclosure, deception, and denial-of-service attacks, respectively. Confidentiality concerns the concealment of data, ensuring it remains known only to the authorized parties. However, in a disclosure attack, the attacker intrudes into the telecommunication network and eavesdrops on the sent message. Although this kind of attack does not have a devastating effect on the system, the attacker may use it to get some knowledge about the system and launch more complicated attacks. Integrity relates to the trustworthiness of data, meaning there is no unauthorized change to the information between the source and destination. In a deception attack, the attacker manipulates the data that is sent through the network and, for example by injecting false data to the measurement signal that is sent to the controller, violates data integrity and deceives the controller to generate the wrong control signal. Availability considers the timely access to information or system functionalities. In a Denial-of-Service (**DoS**) attack, the attacker, by occupying the network bandwidth, prevents the message from arriving at the destination. For instance, the message sent by the plant is actually blocked and does not reach the controller. In this paper, we try to find some solutions for deception attacks.

Various measures to protect cyber-physical systems against cyber attacks can be classified as prevention, detection and mitigation [8]. In prevention, the goal is to prevent the possibility of attacks by reducing the vulnerability of system components, for example by encrypting communication channels, or using firewalls and security protocols [9]. Detection, on the other hand, is an approach in which the system is constantly monitored for anomalies caused by adversary actions, and once an attack is detected, mitigation actions try to reduce the impact of attacks on the system.

There are two important reasons why having detection and mitigation actions is necessary and only prevention actions like encryption are not enough. First of all, there could be a powerful attacker who can break these prevention actions and intrude into the system to establish a malicious attack like what we had before. In recent years, we have had a lot of attacks in different parts of industry, which shows there were some attackers who could break the prevention layer and infiltrate the system. So in this condition we need such detection and mitigation actions to make the system able to tolerate such an attack and remain stable. The second reason to have detection and mitigation actions is that in some systems like power grids, most parts of the equipment are old and implementing prevention measures like encryption will be costly because of the corresponding update of equipment. Therefore, in this case we can use detection and mitigation actions that are completely adaptable to already-implemented industrial control systems. Hence, our aim in this paper is to design methods related to detection and mitigation actions.

In this paper, we propose a novel framework for attack resilient cloud control systems. The framework consists of three parts: an attack detection part to detect anomalies in the system, an attack isolation part to diagnose the location of the attack, and an attack mitigation part to keep the system in a safe mode during and after an attack. Our main contribution in this paper is in the isolation part because, based on the virtual sensor method that we use in the attack mitigation part, we need an isolation method that tells us exactly on which signal(s) there is an attack. So, first we show that other available methods for this objective have some defects such that they have low efficiency and are not applicable to real systems. Then, we propose our novel isolation method that is based on the combination of the concepts of digital twins and cloud computing with control theory. Hence, we make the following novel contributions in this paper:

- Proposing a novel framework for attack resilient cloud control systems.
- Considering having simultaneous attacks on several measurement signals and not only on one of them

- An evaluation of two different methods: observer-based attack detection and an Analytical Redundancy Relations (**ARR**) method for the detection of anomalies in data measured from the sensors in cloud-based industrial control systems
- Proposing a novel isolation method to detect exactly which component(s) have been attacked and we show that it has much better efficiency than the other available isolation method (**ARR**).
- Proposing a mitigation method by developing fault-tolerant control techniques (virtual sensing) for cloud control systems in which we add a reconfiguration block that hides the attack from the controller and makes the controller able to tolerate attack conditions.
- Implementing our proposed security framework on a real testbed as a proof of concept and demonstrating that the detection part can detect attacks in a timely manner, the isolation part can accurately diagnose on which component we have an attack, and the mitigation part can keep the plant stable with acceptable performance during the attack.

The remainder of this paper is organized as follows. Section 2 investigates the related studies and explains the research gap. Section 3 provides a background about CCSs (the real-world system we are studying) and then introduces the real testbed which is used for implementing our proposed framework, and at the end, defines our considered attack model. The proposed solution including attack detection, isolation, and mitigation are explained in Section 4. Section 5 contains all details about our evaluation of the proposed solution. The results of the experiments are given in Section 6. Final remarks and conclusions are discussed in Section 7.

2 RELATED WORKS

Some research has been done regarding detection of deception attacks. Some works have proposed passive attack detection mechanisms by performing some statistical tests on the innovation signal from an estimator [10–12]. The authors in [13] proposed adding watermarking signals to the control inputs and checking received observations by various statistical tests to detect attacks, but adding these watermarking signals can increase the control cost. Also, some studies proposed using Machine Learning (**ML**) algorithms to detect attacks. These algorithms can be employed to learn normal behavior from available data and, then, to compare measured samples to the learned models, to determine if those new samples are anomalous or not. For instance, the authors in [14] proposed using Support Vector Machine (**SVM**), and the authors in [15, 16] proposed a reinforcement learning approach for this purpose.

Once the attack has been detected, a mitigation method is needed to reduce the impact of the attack on the system. Hence, some research has been done regarding mitigation of deception attacks. For example, in [17], the authors have proposed an improved adaptive resilient control scheme for mitigating adversarial attacks such that the controller ensures the asymptotic stability of the closed-loop system and avoids the violation of the state constraints. In [18], a novel data-based adaptive integral sliding-mode control strategy was proposed, which can ensure the stability and a nearly optimal performance of data-driven cyber-physical systems against a class of actuator attacks.

In our prior work [19], we proposed a security framework including detection and mitigation methods for **CCS**s. We demonstrated we can detect attacks in a timely manner using this framework that is deployed in the cloud and once the attack has been detected, an alarm signal is sent to the physical side and makes us able to switch to an ancillary controller to mitigate the attack. In this paper, we have improved our previous work, and instead of employing the ancillary controller we will reconfigure our main controller such that it will be able to control the plant in abnormal state and keep it stable with acceptable performance. Also, in our previous work, we had to send the alarm signal from the cloud to the physical side through a secure communication channel to prevent potential attacks on it. However, in this work, there is no need to send an alarm signal from the cloud to the physical domain and all detection and mitigation actions will be done in the cloud domain.

Since cyber attacks also affect the physical behavior of the system, the tools used for fault-tolerant control can be applied for attack-resilient control. So, here in order to mitigate the attack we reconfigure our controller in the cloud by developing the virtual sensor concept, which is a method to deal with sensor failures. The authors in [14] have also proposed using the virtual sensor concept to mitigate attacks in industrial control systems, especially Energy Management System (**EMS**), but they have skipped the isolation part in this method. Isolation is the necessary and main part in the virtual sensor method that gives knowledge of exactly on which sensor(s) there is an attack. So, in this paper we propose an attack resilient framework for **CCS**s where we develop the virtual sensor concept as a mitigation method, and also we propose a novel isolation method that can exactly diagnose on which sensor(s) an attack has occurred.

In [20], where the virtual sensor concept was first proposed as a fault-tolerant control method to deal with sensor failures, Analytical Redundancy Relations (**ARR**) have also been proposed for isolation. So, we compare our proposed isolation method with this **ARR** method and we show the defects of the **ARR** method and how our method is more powerful than it to diagnose the location of attacks. Furthermore, as an attack detection method in our proposed attack resilient framework, we compare two different methods to detect attacks: observer-based and **ARR**.

3 CLOUD CONTROL SYSTEMS AND ATTACK MODELS

Our targeted system in this paper is cloud control systems and in this section, first we provide a description of cloud control systems then we illustrate our real testbed that we have used to evaluate our proposed security framework. Finally, we specify the attack model that we have considered in this paper.

3.1 BACKGROUND: CLOUD CONTROL SYSTEMS

Figure 1 shows the general structure of cloud control systems. A cloud control system is composed of two layers: the cyber layer and the physical layer. The cyber layer consists of a communication channel and a cloud, while the physical layer contains a plant, actuators and sensors [21]. The plant can be modelled as follows:

$$\begin{aligned} x_{k+1} &= Ax_k + Bu_k + Ed_k \\ y_k &= C_k x_k \end{aligned} \tag{1}$$

where $x \in \mathbb{R}^n$ is the state vector, $y \in \mathbb{R}^p$ is the measurement signal, $u \in \mathbb{R}^{n_u}$ is the control signal, $d \in \mathbb{R}^{n_d}$ is disturbance, A, B, C and E are coefficient matrices, and k is the time instant. In this system the controller is deployed in the cloud, so there is a communication network between the plant and the controller through which the control signals u and the measurement y should be sent. Hence, this communication channel can provide an entry point for attackers to infiltrate the system and manipulate these signals, which can lead to damage and catastrophic consequences. However, under normal conditions in which there is no attack we will have $\tilde{u} = u$ and $\tilde{y} = y$ in Figure 1, and we assume in this normal condition the plant is stable and is controlled well by the cloud controller.

In order to determine how an attack can affect a physical system and jeopardize it, we need to characterize the safety constraints of the system. For this, we use the safe set concept based on [8]. Usually, each physical system has some physical limits: for example in power systems, cables cannot sustain an arbitrarily large instantaneous power. So, based on these limitations and by appropriate scaling of the output of the system y_k using λ, a safe set can be defined for each system as follows:

$$\mathcal{S}_x = \left\{ \mathbf{x} : \max_k \left\{ \|C_x x_k\|_\infty \right\} \leq \lambda \right\} \tag{2}$$

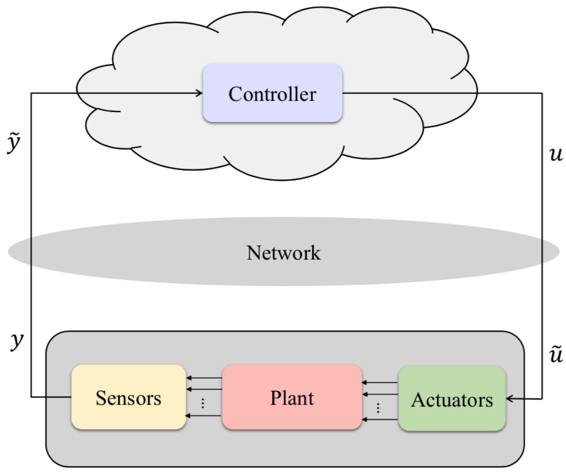


Figure 1: Cloud Control Systems overview.

The system is said to be safe if the state trajectory x_k remains in S_x. Therefore, the attacker, in order to damage the system, tries to drive the state of the system out of its safe set.

3.2 TESTBED DESCRIPTION

As a proof of concept for our proposed security framework, we implemented it on a real testbed whose details can be found in [22].

A. Plant

In our testbed, we use a ball and beam process as the plant. A ball and beam includes a long beam on top of which the ball rolls back and forth. This system is open-loop unstable and the ball swings and falls off the end of the beam. So the controller tries to hold the ball on the setpoint on top of the beam by tilting the beam using an electrical motor. We define the safe set for the ball and beam system with respect to the length of the beam. Since the length of the beam is 1.1 m, the allowed range for the position of the ball is $[-0.55m, 0.55m]$ and the attacker's goal is to drive the ball out of this range and cause the ball to fall off the end of the beam. Also, if the attacker moves the ball from its predefined setpoint but holds it on the beam, it may not damage the system, but can cause extra cost and decrease the efficiency. Hence, our aim in this paper is to hold the ball not only on the beam, but also on the exact setpoint.

We have chosen this system as a plant, because it has a fast dynamic and is time critical and even in the absence of attack, controlling it over the cloud is tricky. Hence, applying our proposed method for this process, and keeping it stable in the presence of attacks can prove the effectiveness of our method very well. The ball and beam system has three measurement signals: the position of the ball y_1, the speed of the ball y_2, and the angle of the beam y_3. This process can be modeled in continuous time as follows:

$$\begin{aligned} \dot{x}(t) &= \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & -\frac{5g}{7} \\ 0 & 0 & 0 \end{bmatrix} x(t) + \begin{bmatrix} 0 \\ 0 \\ 0.44 \end{bmatrix} u(t) \\ y(t) &= \begin{bmatrix} a_1 & 0 & 0 \\ 0 & b_2 & 0 \\ 0 & 0 & c_3 \end{bmatrix} x(t) \end{aligned} \tag{3}$$

where $g = 9.80665$ is the gravity of Earth. We discretize this continuous time model with sampling time 0.05 s for designing the controller and our security framework.

B. Controller

We design an Model Predictive Control (**MPC**) controller to make the ball and beam system stable and control the position of the ball. The control action is obtained by solving, at each sampling instant k, a finite horizon (N) open-loop optimal control problem, using the current state of the plant as the initial state as follows:

$$\begin{aligned} \underset{\mathbf{u}}{\text{minimize}} \quad & J = \sum_{i=k}^{k+N-1} x_i^T Q x_i + u_i^T R u_i + x_{k+N}^T P x_{k+N} \\ \text{subject to} \quad & x_{i+1} = A x_i + B u_i \\ & G \begin{bmatrix} x_i \\ u_i \end{bmatrix} \le g, \quad H \begin{bmatrix} x_i \\ u_i \end{bmatrix} = h \quad , x_{n+k} \in \mathcal{T} \end{aligned} \tag{4}$$

where Q, R and P are cost matrices, A and B define the model of the system, x is the state vector, u is the control signal, and the constraints of the system are defined by the matrices and vectors G, g, H and h. We deployed this controller in a Kubernetes cluster that will be described in the following.

C. Kubernetes cluster

The test-bed has been equipped with a seven-node Kubernetes cluster as the edge cloud. Kubernetes (**K8S**) is a portable, extensible, open-source platform for managing containerized workloads and services that facilitates both declarative configuration and automation [23]. The cluster has been equipped with an nginx ingress [24] and prometheus operator [25]. The nginx ingress is exposed using the **K8S** NodePort paradigm. We use this **K8S** cluster to implement our controller and our attack detection, isolation and mitigation algorithms.

3.3 ATTACK MODEL

In general, cyber-attacks in the literature can be classified into three main types: Denial of Service (**DoS**) attacks, deception attacks, and disclosure attacks [26]. In this paper we consider deception attacks in which the attacker tries to manipulate the data integrity for the transmitted packets between different components of the cyber-physical system. So, in the cloud control systems case, the attacker may manipulate the measurement signal y or control signal u in Figure 1.

Assumption: We consider there is no attack on control signal u and we only have deception attacks on the measurement signal y. However, we consider that is possible to have an attack on several sensor measurements at the same time and we will examine our security framework for all $2^p - 2$ conditions for an attack occurring on y, where p is the number of measurement signals: $y \in \mathbb{R}^p$. We subtract 2 from 2^p, because we disregard the case in which there is no attack on the measurement signals, and also the case in which we have an attack on all measurement signals, since we assume the attacker is not able to have access to all measurement signals at the same time.

By considering the above assumption, in our case, we have three measurement signals in our testbed, and we will consider $2^3 - 2 = 6$ different modes for an attach occurring on the system.

The attacker adds an attack vector $f_a = [a_1 \quad a_2 \quad ... \quad a_p]^T$ to the measurement signal $y = [y_1 \quad y_2 \quad ... \quad y_p]$ and this attack vector has nonzero entries for measurements under attack and zero values for all other measurements. So, we can model this attack using (1) as follows:

$$\tilde{y}_k = C_k x_k + f_a \tag{5}$$

By applying this attack, the controller will receive the manipulated measurement signal and based on that will generate the wrong control signal. This wrong control signal can make the plant unstable and drive the state trajectory of the physical system to an unsafe set that will cause extensive damage to the system.

4 PROPOSED SOLUTION

In this section, we propose a security framework for cloud control systems to ensure the stability of the plant and maintain acceptable performance under

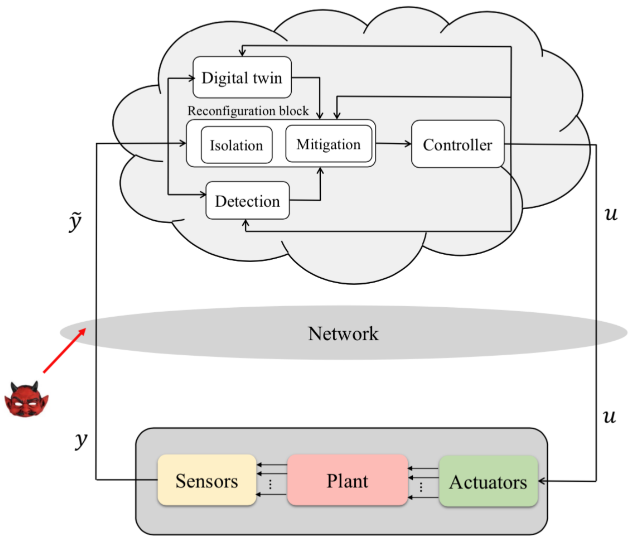


Figure 2: Proposed attack resilient framework overview.

attacks. Actually, using this framework, we detect attacks in a timely manner and then mitigate them to diminish the effects of the attack on the plant. Figure 2 demonstrates an overview of our proposed security framework. As shown in this figure, this framework includes attack detection, isolation, and mitigation parts that all of them are deployed inside the cloud, hence it is adaptable to already implemented **CCS**s' framework, and we do not need to change them a lot. In the following we will explain each part of our framework separately.

4.1 ATTACK DETECTION

In the attack detection part of our proposed security framework, we try to generate a residual signal such that is close to zero and less than a predefined threshold in normal condition during which there is no attack, and it will exceed the threshold once the attack has occurred. In this section we will investigate two different methods to generate residual signals and detect the attack: observer-based and Analytical Redundancy Relations (**ARR**).

A. Observer-based Attack Detection

In this method, we use an observer to estimate the real value of the sensor measurement that has been manipulated by the attacker. The main requirement for using this method is observability of the system. Hence, by assuming

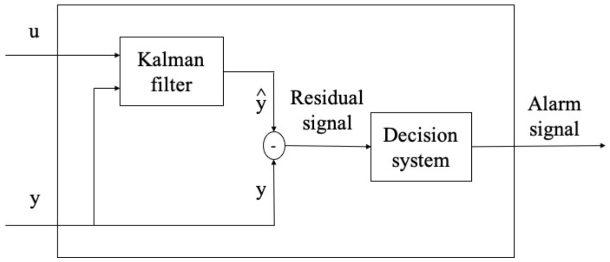


Figure 3: Observer-based attack detection overview.

that our system is observable, we propose designing a Kalman filter as an observer for the system based on our previous work [27]. As you can see in Figure 3, the Kalman filter by using control signals u and measurement signals y tries to estimate the correct value of the sensor measurements $\hat{y}$. Then by comparing y and $\hat{y}$, we can generate a residual signal as follows:

$$r_k = y_k - \hat{y}_k \tag{6}$$

In normal condition, the residual signal should equal to zero, but the measurement noise causes some deviation from zero. Hence,we need a decision function for evaluation of residual signal, and it will determine whether an attack is present. For this, based on our previous work [19, 27] we use a decision function that consists of a test function and a threshold function. Test function $\varphi(r_k)$ provides a measure of the residual's deviation from zero as follows:

$$\varphi(r_k) = |r_k| \tag{7}$$

and this test function will be evaluated by a threshold function $\Phi(k)$ as follows:

$$\begin{cases} \mathcal{H}_0 : \varphi(r_k) \leq \Phi(k) \\ \mathcal{H}_1 : \varphi(r_k) > \Phi(k) \end{cases} \tag{8}$$

where the hypothesis H_0 indicates normal operation of the system and H_1 indicates the abnormal mode of the system that triggers an alarm signal. In this paper, we consider the threshold function a constant value that equals to the absolute value of the maximum deviation of the residual signal from zero in normal condition during which there is no attack.

B. Analytical Redundancy Relations

As is said in Section 4.1-A, observability of the system is naturally required for using observer-based attack detection methods. Analytical redundancy relations are equations that are deduced from an analytical model, which solely use measured variables and control signals as input. The main argument in the **ARR** method is that there is no need to use observer to estimate the unknown states by elimination of these states, so observability of the system is not required in this method. Analytical redundancy relations must be consistent in the absence of an attack, and can thus be used for residual generation. Analytical redundancy can be seen as a tool for obtaining conditions, based on available measurements and control signals, that are necessarily fulfilled when the supervised system works in a normal mode. This method will be designed based on a continuous-time model of the system. Hence, we consider the general continuous-time version of (1) as follows:

$$\begin{aligned} \dot{\boldsymbol{x}}(t) &= \boldsymbol{g}(\boldsymbol{x}(t), \boldsymbol{u}(t), \boldsymbol{d}(t)) \\ \boldsymbol{y}(t) &= \boldsymbol{h}(\boldsymbol{x}(t), \boldsymbol{u}(t), \boldsymbol{d}(t)) \end{aligned} \tag{9}$$

We can determine the nominal and attacked case respectively as provided below:

$$\begin{aligned} \mathcal{H}_0 \Leftrightarrow {} & [\dot{\boldsymbol{x}}(t) = \boldsymbol{g}(\boldsymbol{x}(t), \boldsymbol{u}(t), \boldsymbol{d}(t))] \\ & \wedge [\boldsymbol{y}(t) = \boldsymbol{h}(\boldsymbol{x}(t), \boldsymbol{u}(t), \boldsymbol{d}(t))] \end{aligned} \tag{10}$$

$$\begin{aligned} \mathcal{H}_1 \Leftrightarrow {} & [\dot{\boldsymbol{x}}(t) \neq \boldsymbol{g}(\boldsymbol{x}(t), \boldsymbol{u}(t), \boldsymbol{d}(t))] \\ & \vee [\boldsymbol{y}(t) \neq \boldsymbol{h}(\boldsymbol{x}(t), \boldsymbol{u}(t), \boldsymbol{d}(t))] \end{aligned} \tag{11}$$

where H_0 shows the normal condition and H_1 shows abnormal condition.

To find **ARR**s we will differentiate the output equations q times and q is the minimum natural number that satisfies the following condition:

$$(q+1)p > n + (q+1)n_d \tag{12}$$

Regarding $y \in \mathbb{R}^p$, we have p output equations and by differentiating them q times, we will have $(q+1)p$ equations. Unknown variables in these equations are state variables $x \in \mathbb{R}^n$, disturbance and its differentiation $\bar{d}^{(q)} \in \mathbb{R}^{(q+1)n_d}$. In this paper, $z^{(q)}$ indicate the qth order derivative of variable z, and we have $\bar{z}^{(q)} = [z \quad \dot{z} ... z^{(q)}]^T$. Thus, in order to have enough number of linearly independent equations to calculate the unknown variables based on known variables and eliminate them, we need to start with q times differentiation that q meets (12) and then we need to check the independency of relations, and if there are not $n + (q+1)n_d$ independent equations, we should increase q and differentiate again. Algorithm 1 shows all steps for generating **ARR**s.

Obtained **ARR**s from algorithm 1 can be used for detecting attacks as it has been demonstrated below:

$$
\begin{aligned}
r\left(\bar{\boldsymbol{y}}^{(q)}, \bar{\boldsymbol{u}}^{(q)}\right) &= \mathbf{0} \Leftrightarrow \mathcal{H}_0 \\
r\left(\bar{\boldsymbol{y}}^{(q)}, \bar{\boldsymbol{u}}^{(q)}\right) &\neq \mathbf{0} \Leftrightarrow \mathcal{H}_1
\end{aligned} \tag{13}
$$

Algorithm 1: Algorithm for finding ARRs

```
1  $n$, $p$, $n_d$ and system's model by considering attack vector $\boldsymbol{f_a}$:
     $\dot{\boldsymbol{x}}(t) = \boldsymbol{g}(\boldsymbol{x}(t), \boldsymbol{u}(t), \boldsymbol{d}(t))$
     $\boldsymbol{y}(t) = \boldsymbol{h}(\boldsymbol{x}(t), \boldsymbol{u}(t), \boldsymbol{d}(t)), \boldsymbol{f_a}(t))$
2  ARRs Find the minimum $q$ that satisfies $(q+1)p > n + (q+1)n_d$
3  Find matrix $H^q$ that includes output equations and their differentiation
     up to $qth$ order of derivative:
4  $\begin{bmatrix} \boldsymbol{y} \\ \dot{\boldsymbol{y}} \\ \ddot{\boldsymbol{y}} \\ \vdots \\ \boldsymbol{y}^{(q)} \end{bmatrix} = \begin{bmatrix} \boldsymbol{h}(\boldsymbol{x}, \boldsymbol{u}, \boldsymbol{d}, \boldsymbol{f_a}) \\ \boldsymbol{h}_1\left(\boldsymbol{x}, \bar{\boldsymbol{u}}^{(1)}, \bar{\boldsymbol{d}}^{(1)}, \bar{\boldsymbol{f_a}}^{(1)}\right) \\ \boldsymbol{h}_2\left(\boldsymbol{x}, \bar{\boldsymbol{u}}^{(2)}, \bar{\boldsymbol{d}}^{(2)}, \bar{\boldsymbol{f_a}}^{(2)}\right) \\ \vdots \\ \boldsymbol{h}_q\left(\boldsymbol{x}, \bar{\boldsymbol{u}}^{(q)}, \bar{\boldsymbol{d}}^{(q)}, \bar{\boldsymbol{f_a}}^{(q)}\right) \end{bmatrix} = \boldsymbol{H^q}$
5  while rank $\left(\begin{bmatrix} \frac{\partial \boldsymbol{H}^q}{\partial \boldsymbol{x}} & \frac{\partial \boldsymbol{H}^q}{\partial \bar{\boldsymbol{d}}^{(q)}} \end{bmatrix}\right) \neq n + (q+1)n_d$ do
6      q=q+1
7      Find the new $\boldsymbol{H^q}$ based on step 2 but using the new q
8  if rank $\left(\begin{bmatrix} \frac{\partial \boldsymbol{H}^q}{\partial \boldsymbol{x}} & \frac{\partial \boldsymbol{H}^q}{\partial \bar{\boldsymbol{d}}^{(q)}} \end{bmatrix}\right) = n + (q+1)n_d$ then
9      Use at least the $n + (q+1)n_d$ first equations in $\boldsymbol{H^q}$ to find unknown
         variables $x$ and $\bar{d}^{(q)} = [d \quad d^{(1)} \quad ... \quad d^{(q)}]$ based on known
         variables:
10     $\begin{bmatrix} x \\ \bar{d}^{(q)} \end{bmatrix} = \begin{bmatrix} \phi_x\left(\bar{y}_M^{(q)}, \bar{u}^{(q)}, \bar{f_a}^{(q)}\right) \\ \phi_d\left(\bar{y}_M^{(q)}, \bar{u}^{(q)}, \bar{f_a}^{(q)}\right) \end{bmatrix}$
11     Substitute these obtained variables in the remained equations of $\boldsymbol{H^q}$
         and put $\boldsymbol{f_a}(t) = 0$ to find ARRs:
12     $\mathbf{0} = r\left(\bar{\boldsymbol{y}}^{(q)}, \bar{\boldsymbol{u}}^{(q)}, \mathbf{0}\right)$
```

Using algorithm 1, we can find **ARR**s for our testbed described in Section 3.2 as shown below:

$$r(t) = \begin{bmatrix} r_1(t) \\ r_2(t) \\ r_3(t) \end{bmatrix} = \begin{bmatrix} \dot{y}_1 - \frac{a_1}{b_1} y_2 \\ \dot{y}_2 + \frac{b_2}{c_3} \frac{5g}{7} y_3 \\ \dot{y}_3 - 0.44cu \end{bmatrix} \tag{14}$$

As can be seen, these residual signals are composed of only the outputs y, the outputs' derivatives $y^{(q)}$, and the input u. In normal condition, these residuals should be close to zero and whenever they deviate from zero and exceed the threshold, it demonstrates there is something abnormal in the system.

4.2 ATTACK ISOLATION

Attack isolation means finding on which measurement signals an attack has occurred and determining the location of the attack. We need isolation to know which measurement signals are reliable and we will use this knowledge in the mitigation part. In this section, we provide two different approaches for isolation: **ARR** and our proposed digital twin-based isolation method.

A. Analytical Redundancy Relations

In Section 4.1-B, it was explained how **ARR**s can be used for detecting attacks, and now we want to use them to determine on which measurement signal(s), an attack has occurred. As it can be seen in (13), obtained residuals from **ARR** method are only dependent to measurement signals and its derivatives $\bar{y}^{(q)}$ as well as control signal and its derivatives $\bar{u}^{(q)}$. In **ARR** method, isolation is done based on which residual signal reacts to the attack. Regarding the reaction of residuals to the attacks, we will create a signature for each attack condition and determine on which signals we have an attack. If these residuals react to the attacks on each measurement signal differently, we can diagnose on which measurement signal the attack has occurred.

For our testbed that was described in Section 3.2, based on the residuals that we found for it in (14), r_1 is dependent on $\dot{y}_1$ and y_2, so changes in y_1 and/or y_2 can affect r_1. In the same way, r_2 is dependent on $\dot{y}_2$ and y_3, thus variation in y_2 and y_3 can cause changes in r_2. Finally, r_3 is related to $\dot{y}_3$ and u, therefore manipulation of y_3 and/or u can be reflected in r_3. Based on these relations, we can assign a signature to each attack and do isolation as shown in Table I.

For example, when an attack occurs on y_2, r_1 and r_2 react to this attack. Hence, if we consider $(r_3 r_2 r_1)$ as a binary code, we have $(011)_2 = 3$ that will be signature for attack on y_2. In Table I, we can see each attack has a unique

Table I: Attack isolation using ARRs.

	Attack on y_1	Attack on y_2	Attack on y_3
r_1	1	1	0
r_2	0	1	1
r_3	0	0	1
signature	1	3	6

signature that makes us able to diagnose on which measurement signal the attack has occurred.

Although **ARR**s isolation method seems simple, it is completely dependent on how the model of the system is and also how **ARR**s are related to the measurement signals. Hence, we cannot guarantee that always works. Also, as it said before, in this paper, we consider it is possible to have simultaneous attacks on several measurement signals. consequently, we need an isolation method that can be used for isolation of such simultaneous attacks. However, **ARR** method cannot guarantee that. For example, in our tesbed case, if there are two simultaneous attacks on y_1, and y_2, based on Table I, there will be variation in r_1 due to the attack on y_1, and changes in r_1, and r_2 due to the attack on y_2. Therefore, for simultaneous attack on y_1 and y_2, we totally have changes in r_1 and r_2 and the signature for this attack will be $(011)_2 = 3$ that is exactly same as signature of the single attack on y_2. Regarding these defects of **ARR** isolation method, in next section, we propose a novel isolation method that is able to isolate both single and simultaneous attacks. All other defects of **ARR** isolation method is discussed in Section 6.

B. Proposed digital twin-based isolation method

We need isolation because based on our attack mitigation method that will be explained in Section 4.3, after detecting the attack, in order to be able to mitigate that attack, we will ignore the measurement signals that have been manipulated by the attacker and reconstruct these signals using healthy ones. Hence, by utilizing cloud capacity and based on digital twins concept, we propose a novel attack isolation method to determine which measured signals are under attack.

If we consider we have n sensors, so we will have $2^n - 2$ different modes that the attack can occur on the measurement signals. Thus, in our mitigation part we will design a virtual sensor for each mode to reconstruct the manipulated measurement signals from healthy measurements. As we said before, all three steps: detection, isolation and mitigation are implemented in the cloud, so although virtual sensors do not need complex calculation, we will use the cloud capacity for deploying these $2^n - 2$ virtual sensors.

Figure 4 shows the virtual sensors that each of them have designed for each mode. They use the measurement and control signals and depending on the fact that each virtual sensor works on which mode, it assumes one or some measurement signals are under attack and remove them and use the rest for reconstructing removed signals. Therefore, all $2^n - 2$ virtual sensors try to generate the measurement signals y, but the output of only one of them that has considered the correct mode shows the real value of the measurement signals before manipulating by the attacker. In order to determine which virtual sensor generates the real measurements, we get help from digital twins. Digital twin is a rather new concept in industry. With digital twins, we have virtual replicas of physical systems so that they precisely mirror the internal behavior of the physical systems [28]. Hence, in our isolation method, we will take advantage of digital twins to define a reference operation that we can use for comparing outputs of virtual sensors with it. The signals generated by the virtual sensor that has considered the correct mode will have the minimum difference with the output of the digital twin, and for defining this difference we integrate the absolute error (**IAE**) as follows:

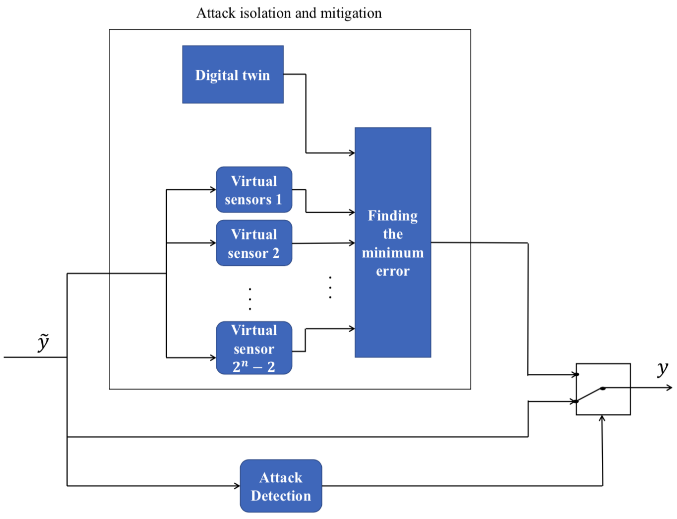


Figure 4: Digital twin-based attack isolation + mitigation

$$E_j = \sum_{i=1}^{k_n} \left| z_i - z_{DT_i} \right|, \quad j \in \{1, 2, ..., 2^n - 2\} \tag{15}$$

where k_n is a window that we use to calculate Integral Absolute Error (**IAE**) from step $k - k_n$ to the current step k, and by doing this we try to consider the history of the differences between out of virtual sensors with output of the digital twin and this will lead to have a more reliable choice than when only calculate the difference at current step k. In (15), z is output of a virtual sensor and z_{DT} is output of the digital twin. So, in each time instant k we calculate $2^n - 2$ errors (**IAE**), and when the alarm signal from the detection part shows there is an attack in system, we choose the virtual sensor that has the minimum error between these $2^n - 2$ calculated error for the mitigation part. Obviously, from this chosen virtual sensor we can realize which signals have been removed and determine the mode of the attack.

In isolation part, the mathematical model of the plant is used For creating digital twin based on [27,29].

4.3 ATTACK MITIGATION

In our previous work [19], we proposed to employ an ancillary controller in physical domain to mitigate impacts of the attack such that once the attack has been detected, we switch from cloud controller to this local controller. We showed that this method works well and we can keep the plant stable under attack. In this paper, our idea for mitigating the attack is reconfiguring the main controller instead of employing a local controller. This idea is adaptable to already implemented **CCS**s' framework, and we do not need to implement a new controller. Hence, it will be more cost efficient and also it can be implemented on more complex system effortlessly.

In our mitigation method, we try to hide the attack from the controller and as you can see in Figure 5, we add a reconfiguration block in the cloud close to the controller and it gets the measurement signal that has been manipulated by the attacker and approximately gives the correct measurement signal to the controller. Therefore, the attacker, whose goal was deceiving the controller and making the system unstable, cannot be successful because the attack will be hidden from controller and the controller will generate correct control signal based on output of reconfiguration block.

In reconfiguration block in Figure 5, in order to reconstruct the measurement signal we utilize the virtual sensor that is also explained in isolation part. In model of the plant (1), each row of matrix C is relate to each sensor measurement. Based on isolation part, we can diagnose that the attack has occurred on which sensors, and, by removing the rows relates to the sensors

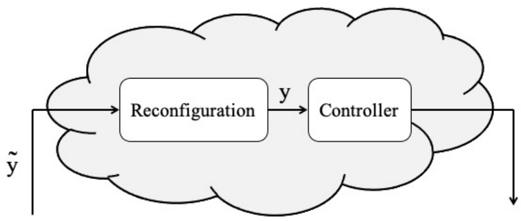


Figure 5: Hiding the attack from the controller using reconfiguration block.

that we have attack on, we can generate matrix C_a. Using this, system with attacks can be described by the state-space model:

$$\begin{aligned} \dot{x}_a &= Ax_a + Bu + Ed \\ y_a &= C_a x_a \end{aligned} \tag{16}$$

where the attacks on sensors is reflected by the matrix C_a. By removing rows related to the sensors under attack from matrix C, and creating C_a, y_a also will contain only healthy measurement signals. To see if real value of other sensor measurement signals that have been manipulated by the attacker, can be reconstructed from y_a, we need to check following condition:

$$\text{Kern}\left(\boldsymbol{C}_a\right) \subseteq \text{Kern}(\boldsymbol{C}) \tag{17}$$

where $Kern()$ denotes the kernel of a matrix. If condition (17) is satisfied, it means (A, C_a) is observable, and the entire state vector can be reconstructed. Regarding this condition we will consider the following assumption.

Assumption: By checking above observability condition for all modes in which the attack may occur on one or several sensors, we consider some redundancy in sensors. We also consider the minimum number of sensors that we need to meet observability condition for all attack modes as protected measurement such that attackers have not access to them.

Now we can design a virtual sensor based on [20] as follows:

$$\begin{aligned} \dot{x}_V &= A_V x_V + B_V u_c + L y_a \\ y_c &= C_V x_V + P y_a \end{aligned} \tag{18}$$

that we have:

$$\begin{aligned} A_V &= A - LC_a \\ B_V &= B \\ C_V &= C - PC_a \\ P &= CC_a^+ \end{aligned} \tag{19}$$

in (19), L is chosen such that $A - LC_f$ is Hurwitz.

5 EXPERIMENTS

To evaluate our proposed security framework, we deploy it on the real testbed that was explained in Section 3.2, and in this section we describe our experiments. In our testbed, we have three sensors for measuring position of the ball, speed of the ball, and angle of the beam. Hence, we can have $2^3 = 8$ different combination of these measurement signal that by disregarding the case in which we have attack on all measurement signals (based on our assumption in Section 3.3 it is not possible), we can define $2^3 - 1 = 7$ different modes as shown in Table II. In this table, mode 1 is related to normal condition in which there is no attack on measurement signals.

Based on (5), and regarding the fact that we have three sensors in our testbed, $f_a = [a_1 \quad a_2 \quad a_3]^T$ is added to measurements signals and depending on that we have attack on which signal(s), a_i can be zero or non zero. In our experiments, we generate these non zero values as follows:

$$a_i = \mathcal{N}\left(\mu, \sigma^2\right), \quad \mu = \lambda_r(k_i - k_0) \tag{20}$$

where σ^2 is constant and μ increases with time. In fact, by applying this attack we will increase real value of the measurement signal gradually with time, such that detecting the attack become difficult. So, The smaller λ_r is, the more difficult it is to detect.

Also, in order to evaluate different parts of our proposed methods in different condition, we utilize Chaos Mesh [30]. Chaos Mesh is an open source cloud-native Chaos Engineering platform. It offers various types of fault simulation and has an enormous capability to orchestrate fault scenarios. Network Chaos is a fault type in Chaos Mesh that we use that for applying

Table II: Different modes of attack.

	Attack on y_1	Attack on y_2	Attack on y_3
mode 1	-	-	-
mode 2	$\times$	-	-
mode3	-	$\times$	-
mode 4	-	-	$\times$
mode 5	$\times$	$\times$	-
mode 6	$\times$	-	$\times$
mode 7	-	$\times$	$\times$

different amount of delay in the Round-Trip Time (**RTT**) between the plant and the controller in the cloud.

5.1 EVALUATION OF ATTACK DETECTION METHODS

In the first part of the experiment, we evaluate the attack detection part. In (20), if λ_r is high, it affects the system and changes the ball's position quickly. However, such an attack will be detected easily. So, the slope should be low and change the ball's position gradually, in which case it is difficult to detect. The length of the beam equals 1.1 meters and the allowed range for the position of the ball is [-0.55 m, 0.55 m]. We chose 0 meters (middle of the beam) as the set-point for the ball's position in the controller. Based on this, choosing $0 < \lambda_r \leq 0.05$ in (20) is reasonable, since it will cause the ball to fall off, and also it will be difficult to detect. So, if we can detect these attacks, we will be able to detect attacks with larger λ_r as well. Hence, for evaluating and comparing two observer-based and ARR-based attack detection methods that we proposed in Section 4.1, for each mode in Table II, we generate attacks based on (20) with each $\lambda_r \in S$ that $S = \{0.001, 0.01, 0.02, 0.03, 0.04, 0.05\}$ and apply it based on (5) on corresponding measurement signals. Then, we compare efficiency of these two different methods for detecting attacks by measuring the time it takes to detect the attack.

5.2 EVALUATION OF ATTACK ISOLATION METHODS

In Section 4.2, we provided two different isolation methods for determining the location of the attack. In the second part of our experiment, we evaluate these isolation method in different modes of attack in which we may have an attack on a measurement signal or a simultaneous attack on several measurement signals, and we show the defects of **ARR** method and effectiveness of our proposed isolation method.

5.3 EVALUATION OF ATTACK MITIGATION METHOD

In the third part of our experiment, we evaluate our mitigation method. For this, we apply different modes of the attack on our test-bed and then we investigate how we can mitigate the attack. For this part, we detect attacks using observer-based attack detection and isolate using our proposed isolation method. Also, as performance metric we use **IAE** as follows to compare the control performance in normal condition, in attack condition when we have mitigation, and in attack condition when we do not have any mitigation.

$$IAE = \sum_{k=0}^{T} |y_k - s_k|, \tag{21}$$

where y_k here is the position signal, and s_k is the set-point for the position of the ball on the beam.

Since in our control system (ball and beam system) the control objective is tracking the reference, we have chosen **IAE** as performance metric for evaluating our mitigation method.

6 RESULTS AND DISCUSSION

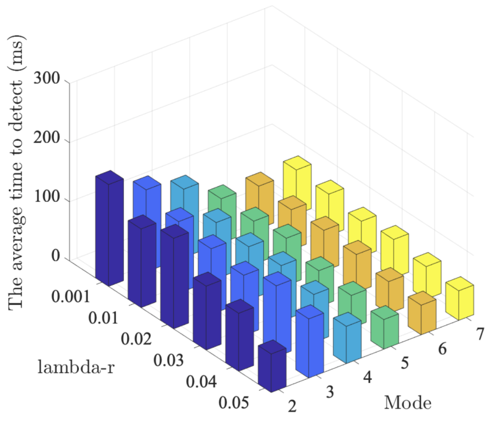
In this section, the results from the experiments detailed in Section 5 are presented.

6.1 ATTACK DETECTION

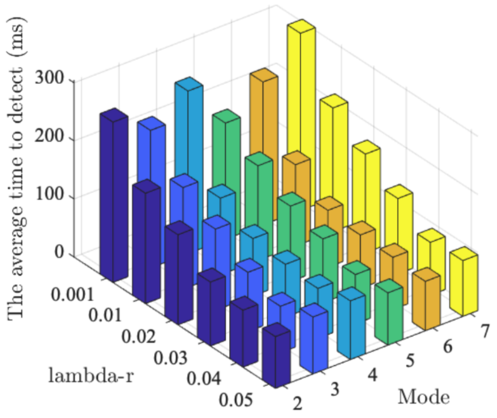
Regarding Section 5.1, for evaluating attack detection part, for each mode of Table II except mode 1 that shows normal condition, we considered attack with different $\lambda_r \in S$. For each mode and each λ_r we run our experiment for 15000 steps and apply the attack on the 14500th sample and then we measure how much time it takes to detect this attack. For each mode and each λ, by applying different delay in **RTT** using Chaos Mesh, we repeat the experiment 10 times each with different **RTT** $\in$ [20.2 ms, 104.1 ms] and calculate the average time it takes to detect the attack. Figure 6 shows the average time it takes to detect the attacks using observer-based and **ARR** attack detection methods. As can be seen, by increasing λ_r the time to detect the attack is decreasing because the steeper the attack signal is, the greater change it makes in the position of the ball and the faster it is detected. So, on average, maximum time for detecting the attack with $\lambda_r = 0.001$ that is the slowest and the most difficult one to detect, is 172.2 ms using observer-based method, and it is 305 ms using **ARR** method. Hence, for other attacks with larger λ_r, it takes less than these time to detect the attack that means both of these attack detection methods can detect attacks really fast. By comparing Figure 6a and Figure 6b it can be seen the time to detect attacks using both methods are close. However, on average, time to detect the attack in each mode using observer-based attack detection is shorter than **ARR** attack detection method.

6.2 ATTACK ISOLATION

In **ARR**-based isolation method, based on Table I, we define the signatures for each mode of attack in Table II for our test-bed in Table III. As can be seen in this table, mode 3 and 5, and also mode 6 and 7 have the same signature. So, when we get signature as 7 we are not able to diagnose that the attack has occurred on angle and position signals or it has occurred on angle and speed signals. Also, when we get signature as 3 we are not able to diagnose



(a) Observer-based detection method's efficiency to detect attacks.



(b) ARR detection method's efficiency to detect attacks.

Figure 6: The average time to detect attacks using observer-based and ARR attack detection methods.

Table III: ARR signatures for each mode of attack.

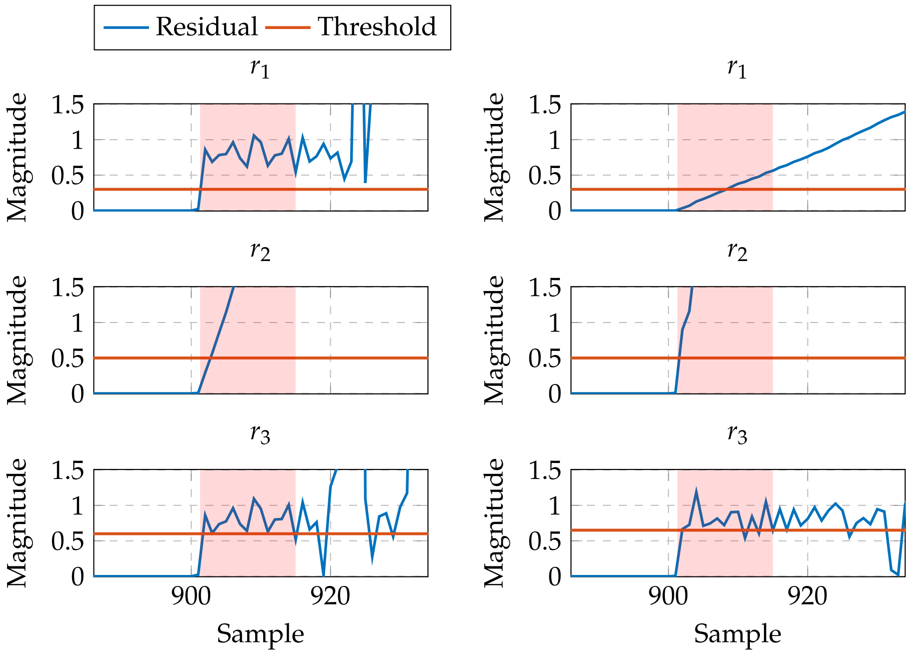
Modes	Mode 2	Mode 3	Mode 4	Mode 5	Mode 6	Mode 7
Signature	1	3	6	3	7	7

that the attack has occurred on both position signal and speed signal or only on speed signal. Figure 7 shows residual signals for ARR_based isolation method for mode 6 and 7. In this figure, all attack are applied on the system at the 900th sample. Figure 7a is related to mode 6, and in this mode, we have simultaneous attacks on position signal and angle signal, and based on Table I, this attack will have effect on r_1 due to the attack on position signal (y_1) and also it will have effect on r_2 and r_3 due to the attack on angle signal (y_3). So, as can be seen in Figure 7a as we expect all residuals exceed their threshold and as signature for this attack we will have $(r_1 r_2 r_3) = (111)_2 = 7$.

Figure 7b is related to mode 7, and in this mode, we have simultaneous attacks on speed signal and angle signal. based on Table I, this attack will have effect on r_1 and r_2 due to the attack on speed signal (y_2) and also it will have effect on r_2 and r_3 due to the attack on angle signal (y_3). So, as can be seen in Figure 7b as we expect all residuals exceed their threshold and as signature for this attack we will have $(r_1 r_2 r_3) = (111)_2 = 7$. So, for both cases mode 6 and 7 we got same signature 7 and that means that if we get signature 7, we will not be able to distinguish we are in mode 6 or 7.

Therefore, one of the main weak point of ARR-based isolation method is that this method is completely dependent on model of the plant and based on that signature for different modes of attack will be defined and we do not have any control on that. Hence, we may have similar signature for different modes and not to be able to diagnose the correct mode of attack as we had this problem for our testbed in Table III. However, we do not have such problem in our proposed isolation method since we design an observer for each mode and calculate the error for each mode separately.

In addition to this problem, there are also two critical issues in ARR-based isolation method. The first one that has big impact on correct isolation, is defining an appropriate threshold such that if the residual signal exceeds this threshold we can consider it as one for generating signature, otherwise it will be considered as zero. The second issue is related to this fact that residual signals that will generate the signature of the attack do not exceed their threshold at the same time. Therefore, we need to define a window that shows the certain amount of time that we should wait after that the first residual exceeds its threshold to see which other residuals will exceed their threshold to consider them as one and the rest as zero and decide about the signature for the attack. Because after a while that the plant is going to be unstable due to the attack, all residuals will start to increase and they may exceed their threshold. Hence, we should consider only the residuals that exceed their threshold inside the window. Red area in Figure 7 and 8 demarks the window time.



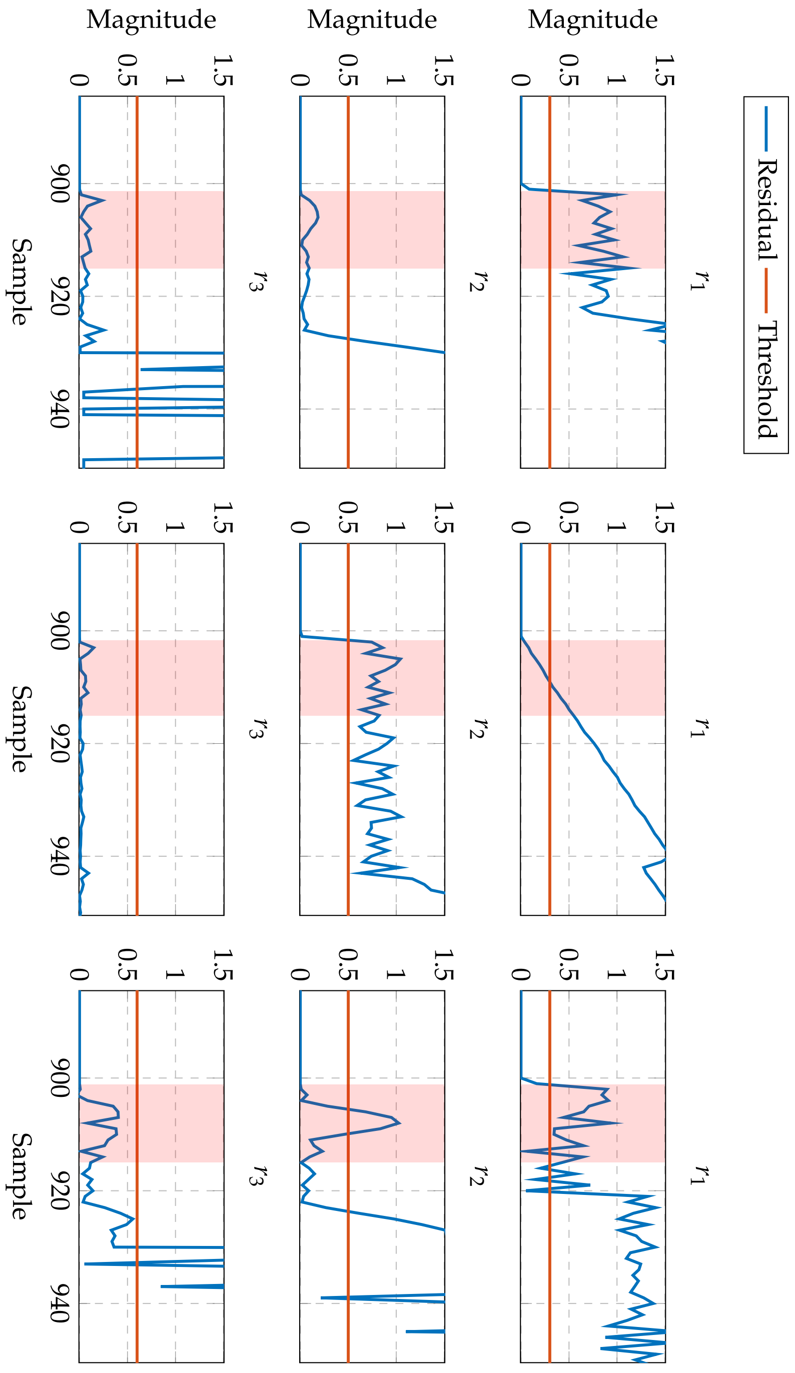
(a) Applying attacks on both position and angle signals (Mode6).

(b) Applying attacks on both speed and angle signals (Mode7).

Figure 7: Residual signals for ARR_based isolation method for mode 6 and 7.

Choosing thresholds for residual signals and window time is difficult because it should work for all conditions. Smaller threshold will lead to the residual signal exceeds the threshold and we will have faster isolation, but it will also cause some false alarms and the residuals that should be considered as zero will be considered as one and consequently we will have wrong signature and wrong isolation. Longer window time is more conservative and causes not to miss the residuals that will exceed their thresholds a bit later. However, longer window time will lead to wait more and consequently it takes more time to do isolation and this will affect on attack mitigation. Because, if the attack is powerful, it will make the plant unstable soon and we need to detect, isolate and mitigate this attack as fast as possible to save the system.

Figure 8a shows residual signals for the condition in which we have an attack on position signal at the 900th sample. So, in this condition based on Table I, we expect only r_1 exceed its threshold to create signature $(001)_2 = 1$. Regarding our chosen thresholds in Figure 8a, during the window time only



(a) Applying attacks on position signal (Mode2).

(b) Applying attacks on speed signal (Mode3).

(c) Applying attacks on position signal (Mode2) with delay.

Figure 8: Threshold challenges in ARR_based isolation method

r_1 exceeds its threshold as we expected. Figure 8b also shows residual signals for condition we have an attack on speed signal at the 900th sample. So, it seems chosen thresholds work well and as we expect based on Table I and III, during the window time r_1 and r_2 exceed their threshold. However, in Figure 8c residual signals for condition we have an attack on position signal at the 900th sample and in this condition, we have also applied 40 ms delay using Chaos Mesh such that the average **RTT** in this condition is about 66 ms. In this condition, same as Figure 8a, we expect during window time only r_1 exceed its threshold, but we can see r_2 also has exceeded its threshold that will lead to have wrong signature and wrong isolation.

For solving such problem we can either increase threshold for r_2 or decrease window time, but both of these changes are so challenging. For example here if we want to increase the threshold for r_2 to solve the problem in Figure 8c such that r_2 that has passed the threshold remains under threshold, it will cause a problem in Figure 8b since in this figure r_2 is supposed to surpass the threshold, but if we increase the threshold such that r_2 in Figure 8c remains under threshold, r_2 in Figure 8b also will be so close to the threshold or lower than it that will cause misleading and generating wrong signature.

On the other hand, decreasing window time for solving the problem in Figure 8c, cannot be an option. Since if the window time is decreased such that the time when r_2 exceeds the threshold is out of window time and is not considered for generating signature, this may cause missing the residuals that will exceed their thresholds a bit later. For example, r_1 in Figure 8b takes more time to exceed its threshold, so we need to consider not too short window time for considering such residual as one.

Increasing both threshold and window time together could be a solution for this problem, but it will lead to wait more and consequently it takes more time to do isolation and this will affect on attack mitigation. For example, this can solve the problem of r_2 in Figure 8c, but in Figure 8b, we should wait about 40 sampling steps to consider r_2 as one for generating signature that is too long and does not work for the ball and beam process that has fast dynamic and makes it unstable.

Therefore, ARR-based isolation method not only does not work for attacks that cause same signature, but also choosing appropriate threshold and window time is really challenging and has big impact on the result. Also, delay can affect the result and cause generating wrong signature and wrong isolation. So, **ARR** may work for detection, but it really has low efficiency as isolation method. Because in attack detection using **ARR**, we care only about the first residual signal that exceeds its threshold, but for isolation all residual signals should be considered.

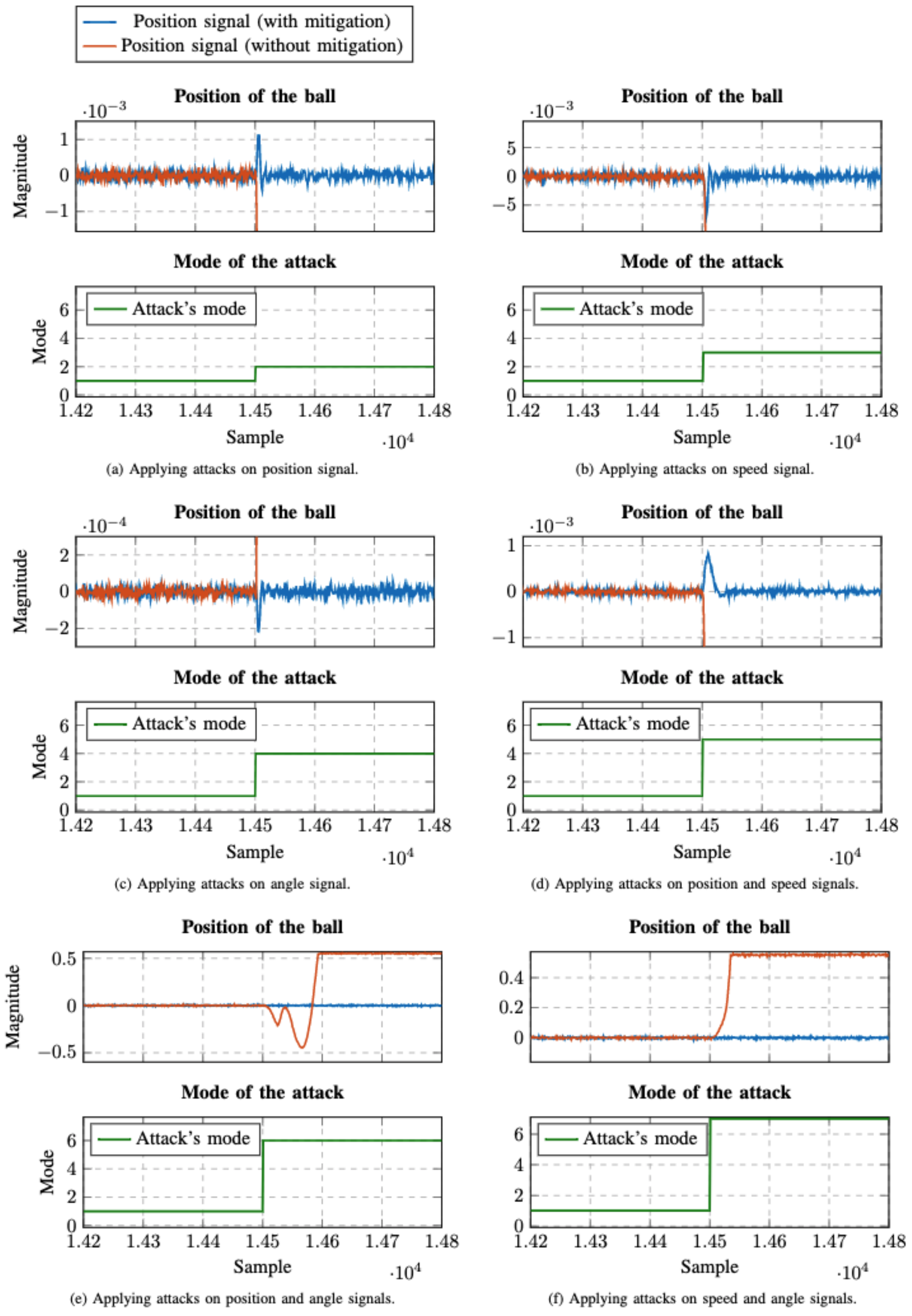


Figure 9: Isolation using our proposed method and mitigation based on this isolation.

Figure 9 shows performance of our proposed isolation method to determine on which measurement signals the attack has occurred and specify the mode of attack. In this figure, we generate the attack based on (20) with $\lambda_r = 0.04$ and apply it on related measurement signals to each mode of Table II at the 14500th sample. For instance, in mode 2 we apply this attack only on position signal, and in mode 6 we apply it on both position and angle signals. Also, using Chaos Mesh, we apply network delay 40 ms such that the average **RTT** between the plant and the controller in the cloud is about 66.1 ms in these experiments. As can be seen in Figure 9, in all six modes of attack, even in the presence of applied delay in **RTT**, our proposed isolation method works well and after detecting the attack by observer-based attack detection part, can exactly specify the mode of the attack and diagnose on which sensor there is an attack.

6.3 ATTACK MITIGATION

In the following, based on the detection and isolation from previous section, we activate our mitigation part to mitigate the impact of the attack. In Figure 9, we can also see the position signal for each mode of attack. As can be seen in the figure, blue curve shows the position of the ball on the beam in the presence of our mitigation method, and the red curve shows the position of the ball on the beam in the absence of the mitigation method. For example, in Figure 9d, the attack applies on position and speed signals simultaneously at the 14500th sample, and starts to cause the ball to deviate from its set-point in order to make it off the beam. However, this attack is detected by the observer-based attack detection at the 14503th sample and activates our proposed isolation method. Then, our proposed isolation method specify mode 5 for this attack that based on Table II means that there are attacks on position and speed signal. Based on this isolation, in our mitigation part, in reconfiguration block in Figure 5, before feeding measurement signals to the controller, position and speed signals are removed and regenerated using rest of measurement signals and then these new signals are fed to the controller. By doing this, as blue curve in Figure 9d shows, our mitigation method moves the ball back to the set-point. Otherwise, in the absence of this mitigation, the ball continues to deviate from the set-point following the red curve in Figure 9d, and at the end it will fall off from the end of the beam.

Regarding Section 5.3, in order to evaluate our mitigation method, and to see if it can keep the system stable with an acceptable performance, we measure the performance of the controller using **IAE**. Figure 10 shows **IAE** for normal condition during which there is no attacks, attack condition without mitigation, and attack condition with our mitigation in each attack mode. In all cases the **IAE** is measured up until the point where the ball falls off the

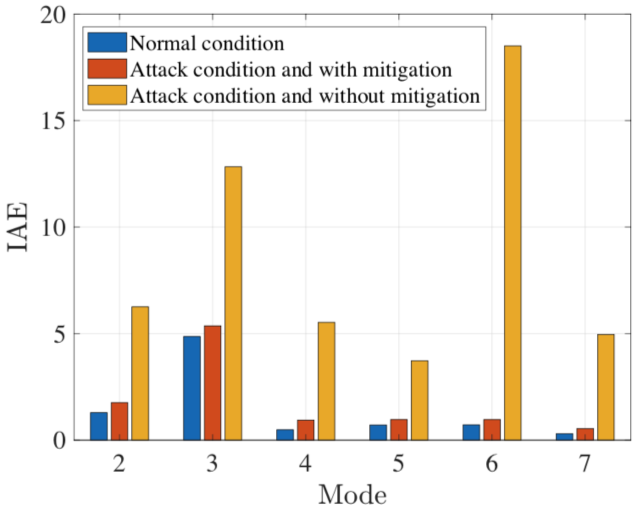


Figure 10: IAE for each mode of system.

beam. As can be seen in this figure, in all modes of the attack **IAE** for the condition that we mitigate the attack using our proposed security framework is close to **IAE** in normal condition that this proves that we can keep the plant stable with an acceptable performance during the attack.

7 CONCLUSIONS

In this work, an attack resilient framework for cloud control systems has been proposed, and its effectiveness has been proved by implementing it on a real cloud-based testbed. Two observer-based and **ARR** methods were investigated and evaluated as attack detection in this framework. We showed both of these two methods have acceptable performance, and able to detect attacks fast, but the observer-base method is able to detect attacks in a shorter time.

In the isolation part, first we evaluated the available isolation method **ARR** and showed that ARR-based isolation method not only does not work for attacks that cause same signature, but also choosing appropriate threshold and window time is really challenging and has big impact on the result. Also, delay can affect the result and cause generating wrong signature and wrong isolation. So, **ARR** may work for detection, but it really has low efficiency as isolation method. Regarding defects of this method, a novel method by combination of digital twin concept, cloud computing, and control theories.

We showed that in comparison to **ARR**, it has a promising performance, and does not have the flaws that are raised about **ARR** as an isolation method. This method is able to diagnose the mode of attack correctly, and delay in **RTT** does not affect its performance.

Our novel isolation method uses the concepts of digital twins and cloud computing, and that is a departure from previous methods, which usually are very much based in pure control theory, and gives a whole new way to approach these types of problems, that ties into current hot research trends.

We also proposed a mitigation part in this framework by developing the virtual sensor concept for cloud control systems from fault-tolerant control system. By applying different modes of attack on the system we proved that this mitigation method can keep the system stable with an acceptable performance during the attack. So, even if the attacker can break the prevention scenarios and intrude into the system to establish an attack, we can make the system able to telerate this attack using our proposed framework.

Future work to investigate other kinds of attacks on **CCS**s like Replay attack will be carried out to design some methods to detect and mitigate these kinds of attacks. Also, we will study some new methods for delay compensation between the cloud and the plant. In this paper, we used **MPC** controller for this objective, and using its predictive features we tried to deal with delay problem. As a future work, we will design delay compensation methods that make us able to have even simpler controllers inside the cloud instead of **MPC**.

Bibliography

[1] Y. Xia, "Cloud control systems," *IEEE/CAA Journal of Automatica Sinica*, vol. 2, no. 2, pp. 134–142, 2015.

[2] R. Langner, "Stuxnet: Dissecting a cyberwarfare weapon," *IEEE Security & Privacy*, vol. 9, no. 3, pp. 49–51, 2011.

[3] D. Alert, "Cyber-attack against ukrainian critical infrastructure," *Cybersecurity Infrastruct. Secur. Agency, Washington, DC, USA, Tech. Rep. ICS Alert (IR-ALERT-H-16-056-01)*, 2016.

[4] ICS-CERT, "Hatman—safety system targeted malware," Mar. 2017. [Online]. Available: https://ics-cert.us-cert.gov/MAR-17-352-01-HatManTargeted-Malware.

[5] Kaspersky Lab ICS-CERT, "Threat landscape for industrial automation systems in h2 2017," Mar. 2018. [Online]. Available: https://icscert.kaspersky.com/reports/2018/03/26/threat-landscape-for-industrial-automation-systems-in-h2-2017/.

[6] N. S. Malik, R. Collins, and M. Vamburkar, "Cyber-attack, pings data systems of at least four gas networks," Apr. 2018. [Online]. Available: https://www.bloomberg.com/news/articles/2018-04-03/day-after-cyberatta ck-a-third-gas-pipeline-data-system-shuts.

[7] M. A. Bishop, "The art and science of computer security," 2002.

[8] A. Teixeira, K. C. Sou, H. Sandberg, and K. H. Johansson, "Secure control systems: A quantitative risk management approach," *IEEE Control Systems Magazine*, vol. 35, no. 1, pp. 24–45, 2015.

[9] O. Vukovic, K. C. Sou, G. Dan, and H. Sandberg, "Network-aware mitigation of data integrity attacks on power system state estimation," *IEEE Journal on Selected Areas in Communications*, vol. 30, no. 6, pp. 1108–1118, 2012.

[10] F. Pasqualetti, F. Dörfler, and F. Bullo, "Attack detection and identification in cyber-physical systems," *IEEE transactions on automatic control*, vol. 58, no. 11, pp. 2715–2729, 2013.

[11] E. Mousavinejad, F. Yang, Q.-L. Han, and L. Vlacic, "A novel cyber attack detection method in networked control systems," *IEEE transactions on cybernetics*, vol. 48, no. 11, pp. 3254–3264, 2018.

[12] X. Ge, Q.-L. Han, M. Zhong, and X.-M. Zhang, "Distributed krein space-based attack detection over sensor networks under deception attacks," *Automatica*, vol. 109, p. 108557, 2019.

[13] Y. Mo, S. Weerakkody, and B. Sinopoli, "Physical authentication of control systems: Designing watermarked control inputs to detect counterfeit sensor outputs," *IEEE Control Systems Magazine*, vol. 35, no. 1, pp. 93–109, 2015.

[14] K. Paridari, N. O'Mahony, A. E.-D. Mady, R. Chabukswar, M. Boubekeur, and H. Sandberg, "A framework for attack-resilient industrial control systems: Attack detection and controller reconfiguration," *Proceedings of the IEEE*, vol. 106, no. 1, pp. 113–128, 2017.

[15] M. N. Kurt, O. Ogundijo, C. Li, and X. Wang, "Online cyber-attack detection in smart grid: A reinforcement learning approach," *IEEE Transactions on Smart Grid*, vol. 10, no. 5, pp. 5174–5185, 2018.

[16] D. An, Q. Yang, W. Liu, and Y. Zhang, "Defending against data integrity attacks in smart grid: A deep reinforcement learning-based approach," *IEEE Access*, vol. 7, pp. 110 835–110 845, 2019.

[17] L. An and G.-H. Yang, "Improved adaptive resilient control against sensor and actuator attacks," *Information Sciences*, vol. 423, pp. 145–156, 2018.

[18] X. Huang, D. Zhai, and J. Dong, "Adaptive integral sliding-mode control strategy of data-driven cyber-physical systems against a class of actuator attacks," *IET Control Theory & Applications*, vol. 12, no. 10, pp. 1440–1447, 2018.

[19] F. Akbarian, W. Tärneberg, E. Fitzgerald, and M. Kihl, "A security framework in digital twins for cloud-based industrial control systems: Intrusion detection and mitigation," in *2021 26th IEEE International Conference on Emerging Technologies and Factory Automation (ETFA)*. IEEE, 2021, pp. 01–08.

[20] M. Blanke, M. Kinnaert, J. Lunze, M. Staroswiecki, and J. Schröder, *Diagnosis and fault-tolerant control*. Springer, 2016, vol. 2.

[21] Z. Xu and Q. Zhu, "Secure and resilient control design for cloud enabled networked control systems," in *Proceedings of the first ACM workshop on cyber-physical systems-security and/or privacy*, 2015, pp. 31–42.

[22] F. Akbarian, W. Tärneberg, E. Fitzgerald, and M. Kihl, "A cloud-control system equipped with intrusion detection and mitigation," *Electronic Communications of the EASST*, vol. 80, 2021.

[23] Kubernetes. [Online]. Available: https://kubernetes.io

[24] Ingress-nginx. [Online]. Available: https://github.com/kubernetes/ingress-nginx

[25] Prometheus-operator. [Online]. Available: https://github.com/coreos/prometheus-operator

[26] M. S. Mahmoud and Y. Xia, *Cloud Control Systems: Analysis, Design and Estimation*. Academic Press, 2020.

[27] F. Akbarian, E. Fitzgerald, and M. Kihl, "Intrusion detection in digital twins for industrial control systems," in *2020 International Conference on Software, Telecommunications and Computer Networks (SoftCOM)*. IEEE, 2020, pp. 1–6.

[28] M. Farsi, A. Daneshkhah, A. Hosseinian-Far, H. Jahankhani *et al.*, *Digital twin technologies and smart cities*. Springer, 2020.

[29] A. Moser, C. Appl, S. Brüning, and V. C. Hass, "Mechanistic mathematical models as a basis for digital twins," in *Digital Twins*. Springer, 2020, pp. 133–180.

[30] Chaos mesh. [Online]. Available: https://chaos-mesh.org/