IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received 21 February 2023, accepted 8 March 2023, date of publication 20 March 2023, date of current version 23 March 2023.

Digital Object Identifier 10.1109/ACCESS.2023.3259063

RESEARCH ARTICLE

Attack Resilient Cloud-Based Control Systems for Industry 4.0

FATEMEH AKBARIAN, (Member, IEEE), WILLIAM TÄRNEBERG, (Member, IEEE), EMMA FITZGERALD, (Member, IEEE), AND MARIA KIHL, (Member, IEEE)
Department of Electrical and Information Technology, Lund University, 22100 Lund, Sweden

Corresponding author: Fatemeh Akbarian (fatemeh.akbarian@eit.lth.se)

This work was supported in part by the Celtic-Next Project IMMINENCE and the SSF Project SEC4FACTORY under Grant SSF RIT17-0032, in part by the Excellence Center at Linköping-Lund on Information Technology (ELLIIT) Strategic Research Area, and in part by the Nordic University Hub on Industrial Internet of Things (HI2OT) funded by NordForsk. The work of Maria Kihl was supported in part by the Wallenberg Artificial Intelligence (AI), Autonomous Systems and Software Program (WASP) funded by the Knut and Alice Wallenberg Foundation.

ABSTRACT In recent years, since the cloud can provide tremendous advantages regarding storage and computing resources, the industry has been motivated to move industrial control systems to the cloud. However, the cloud also introduces significant 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 high performance during the attack.

INDEX TERMS Attack detection, attack isolation, attack mitigation, cloud control systems, resilient control.

I. INTRODUCTION

In recent years, we have had numerous advances in network technology, and these technologies have been combined with control systems to create network control systems (NCS). In this type of control system, the control loop is closed through the communication channel, making monitoring and adjusting the plant remotely possible. Control systems usually have to deal with big data. 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. These problems can have a significant negative impact on the industry. Industry 4.0 is the next generation of the industry that focuses heavily on interconnectivity, automation, and real-time data and makes great efforts to cope with these problems.

Industry 4.0 integrates industry with some modern technologies, including the Internet of Things (IoT), cloud computing, data analytics, Artificial Intelligence (AI), and machine learning into their production facilities and throughout their operations [1]. 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 [2]. Internet of Things (IoT) can be described as a communication infrastructure and methodology between objects [3]. These objects can be provided with sensors or actuators and easily make information available or perform complex actions. Industrial Internet of Things (IIoT) is a subcategory of IoT and is used for industrial purposes such as manufacturing, monitoring, and supply chain management. Heterogeneous industrial Internet of things (Het-IoT) is also

The associate editor coordinating the review of this manuscript and approving it for publication was Nitin Gupta.

introduced based on the key characteristic of IoT: heterogeneity. In Het-IoT, it is vital that machines with different hardware platforms and networks can have efficient functioning and interaction [4].

IoT devices generate a huge amount of data that must be stored somewhere. One of the important technologies in industry 4.0 is cloud computing which provides the storage resources and processing power needed to make use of this data. Hence, by getting advantages of introduced technologies in industry 4.0 and combining IoT and cloud computing, issues of resource-constrained NCSs can be almost solved. By combining the benefits of network control and cloud computing technology, a new concept called cloud control system (CCS) has been developed. In CCSs, the core processing unit is shifted to a cloud server and endows the control system with massive parallel computation [5].

Industry 4.0 is technology-driven and is based on the digital transformation of the industry. Based on the assumption that Industry 4.0 focuses less on the original principles of social fairness and sustainability but more on digitalization and AI-driven technologies for increasing the efficiency and flexibility of production, Industry 5.0 was introduced [6]. Industry 5.0 is value-driven and is based on three pillars i.e., human-centric, sustainability, and resilience [7]. Industry 5.0 aim to increase the degree of personalization and focuses on making industries more robust, intelligent, and smarter.

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 drive revolutionary changes to connected devices, it also brings with it new cyber risks for which the industry is unprepared [8]. Although combining the cloud with control systems has many benefits, 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. This communication can be exposed to various security attacks, including passive and active attacks. In recent years, several attacks have targeted control systems and caused damage [9], [10], [11], [12], [13]. This indicates the possibility of such attacks on CCSs and the need for appropriate security measures to protect these systems.

To see how cyber-attacks can affect systems, computer security literature identifies three fundamental properties of information and services in IT systems, namely confidentiality, integrity, and availability, often denoted as CIA, and they can be violated by disclosure, deception, and denial-of-service attacks, respectively. In this paper, we try to find a solution for deception attacks that target data integrity in cloud control systems. 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 sent through the network. For example, by injecting false data into the measurement signal sent to the controller violates data integrity and deceives the controller into generating the wrong control signal.

Various measures to protect systems against cyber-attacks can be classified as prevention, detection, and mitigation [14]. 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 [15]. On the other hand, detection is an approach in which the system is constantly monitored for anomalies caused by adversary actions. 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 the industry, which shows some attackers 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 [16]. 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.

For attack mitigation, we get the advantage of the virtual sensor concept, which is a method to deal with sensor failures. In this method, the controller is reconfigured when a fault is detected by removing the faulty sensors' data and reconstructing them based on healthy ones [17]. This method has a simple algorithm that does not have any complex computation that takes time, and also, implementing it does not need to add any new equipment. Hence, it is beneficial to develop this method for mitigating attacks' effects. However, there is a tricky requirement for this method that makes using it difficult. In fact, the main requirement of the virtual sensor technique is isolation which means diagnosing exactly on which sensor(s) there is an attack. As we will survey in section II, the papers that have proposed virtual sensor method for attack mitigation either have skipped the isolation part [18] or their proposed isolation method has some defects such that they have low efficiency and are not applicable to real systems that we will explain in section VI-B [17].

Therefore, in this paper, we propose a novel isolation method that is based on the combination of the concepts of digital twins and cloud computing with control theory and makes it possible to develop virtual sensors method for attack mitigation.

To show how this isolation method works and evaluate it, first, we need to have a method to detect the attack and then show that our isolation method can diagnose the location of the detected attack. Then, using knowledge from the isolation part, we try to mitigate the attack's effect using the virtual sensor method. Hence, we introduce a novel framework consisting 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. Hence, we make the following novel contributions in this paper:

- Proposing a novel framework for attack-resilient cloud control systems by introducing a new isolation method.
- An evaluation of two different methods: observer-based attack detection and an analytical redundancy relation (ARR) method for detecting 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 even in the presence of simultaneous attacks on several measurement signals, and we show this method has much better efficiency than the other available isolation method (ARR).
- Proposing a mitigation method by developing fault-tolerant control techniques (virtual sensors) 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 good performance during the attack.

The remainder of this paper is organized as follows. Section II investigates the related studies and explains the research gap. Section III provides 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, is explained in Section IV. Section V contains all details about our evaluation of the proposed solution. The results of the experiments are given in Section VI. Final remarks and conclusions are discussed in Section VII.

II. RELATED WORK

Recently, regarding the increasing number of attacks in industry, many researchers have been attracted to this problem, and some studies have been done.

Authors of [3] have proposed a method to detect the threats in IIoT based on Hidden Markov Model (HMM). In this method, HMM is used to model sequential data which is generated from IIoT devices. A Genetic Algorithm (GA) is applied to optimize the parameters of HMM. Also, a dynamic window-based sequence extractor has been proposed to extract multiple sequences simultaneously before processing by multi-HMM.

In [19], an anomaly detection method has been proposed for IIoT named ASTREAM, which can accomplish efficient and accurate anomaly detection with good scalability. This method merges the sliding window, change detection, and model update strategies into LSHiForest, and it can effectively handle the infiniteness, correlations, and distribution change of data streams.

Trust affects the consumption pattern of a specific service that is provided by an IIoT device. However, due to the lack of perception in machines, trust cannot be built especially since each object is interpreted differently and different applications running on the IIoT devices may assign different trust scores. Hence, the authors of [7] first propose trust metrics. Then, they present a trust model based on the neutrosophic weighted product method (WPM) used by IIoT applications to assess IIoT devices' trust scores. The developed model assesses devices' trustworthiness based on the spatial knowledge, temporal experience, and behavioral patterns retrieved from the IIoT devices. Finally, they use neutrosophic K-NN clustering and neutrosophic support vector machines (SVM) to classify the extracted characteristics to generate the final trust score and make a decision.

Generally, available methods for attack detection can be divided into two different groups: model-based methods, like designing estimators, and data-based methods, like using machine learning methods. Here, we survey some of these methods.

Some studies have proposed using machine learning (ML) algorithms to detect attacks. These algorithms can be employed to learn normal behavior from available data and then compare measured samples with these learned models to determine if that is anomalous or not. In [20], a review of recently proposed deep learning (DL) solutions for detecting cyber-attacks has been provided that shows DL modules can be used to detect cyber-attacks.

Also, some studies have proposed model-based methods. In [21], a distributed filtering algorithm is proposed to estimate the system state, and an attack detector is designed by considering a dynamic threshold. The authors of [22] proposed adding watermarking signals to the control inputs and checking received observations by various statistical tests to detect attacks. However, adding these watermarking signals can increase the control cost. In this paper, they tried to reduce the control cost when the system is not under attack. The authors of [23] also proposed adding the watermarking signal to control input by designing a dual-rate control framework, including a model predictive controller and a state-feedback predictor-based controller. Also, they consider a reconfiguration block to mitigate the effects of the watermarking signal on control costs.

One of the most important and common model-based methods used for anomaly detection from years ago until

today is based on the concept of designing an estimator. In this method, an estimator such as Kalman filter is designed to estimate the system's state, and the real value of the measured signal is estimated based on that. Then, by comparing the measured signals that we get from sensors with the estimated ones, a residual signal is generated, and by performing statistical tests such as generalized likelihood ratio (GLR) or cumulative sum (CUSUM) on this signal, an anomaly is detected. This anomaly detection method has been applied to different applications like power systems [24], [25], [26], [27], automated vehicles [28], [29], [30], industrial control systems [31], [32], [33], etc.

Once the attack has been detected, a mitigation method is needed to reduce the attack's impact on the system. Hence, some research has been done regarding the mitigation of deception attacks. For example, in [34], 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 [35], a novel data-based adaptive integral sliding-mode control strategy was proposed, which can ensure the stability and nearly optimal performance of data-driven systems against a class of actuator attacks. The authors of [31] have proposed a secure control design for mitigating false data injection attacks. This includes a robust controller that considers this kind of attack as model uncertainties. At the same time, it compensates for measurement noise and process noise.

In our prior work [36], we proposed a security framework including detection and mitigation methods for CCSs in Industry 4.0. We demonstrated that we could detect attacks in a timely manner using this framework that is deployed in the cloud. Once the attack has been detected, an alarm signal is sent to the physical side, which 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 an abnormal state and keep it stable with good 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 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 of [18] have also proposed using the virtual sensor concept to mitigate attacks in industrial control systems, especially energy management systems (EMSs), but they have skipped the isolation part in this method. Isolation is the necessary and main part of 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 CCSs 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 [17], 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 is 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.

III. CLOUD CONTROL SYSTEMS AND ATTACK MODELS

Our targeted system in this paper is cloud control systems which is one of the main technologies of Industry 4.0. Cyber-physical systems (CPS) integrate sensing, computation, control, and networking into physical objects and infrastructure, connecting them to the internet and each other. A cloud control system is a specific type of cyber-physical system in which the controller is deployed in 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. In this section, we first describe cloud control systems and then illustrate the real testbed we have used to evaluate our proposed security framework. Finally, we specify the attack model that we have considered in this paper.

A. BACKGROUND: CLOUD CONTROL SYSTEMS

Fig. 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 [37]. The plant can be

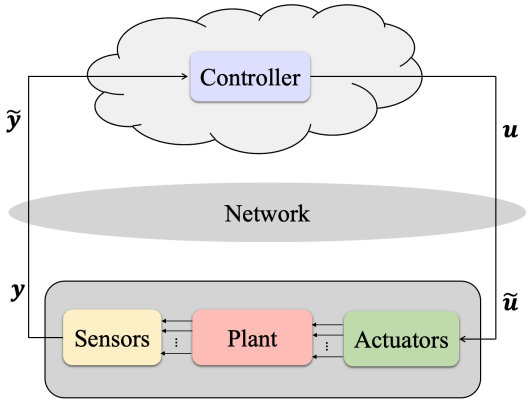


FIGURE 1. Cloud Control Systems overview.

modeled as follows:

$$\begin{aligned} \boldsymbol{x}(k+1) &= \boldsymbol{A}\boldsymbol{x}(k) + \boldsymbol{B}\boldsymbol{u}(k) + \boldsymbol{E}\boldsymbol{d}(k) \\ \boldsymbol{y}(k) &= \boldsymbol{C}\boldsymbol{x}(k) + \boldsymbol{v}(k) \end{aligned} \tag{1}$$

where $\boldsymbol{x} \in \mathbb{R}^n$ is the state vector, $\boldsymbol{y} \in \mathbb{R}^p$ is the measurement signal, $\boldsymbol{u} \in \mathbb{R}^{n_u}$ is the control signal, $\boldsymbol{d} \in \mathbb{R}^{n_d}$ is disturbance, $\boldsymbol{v} \in \mathbb{R}^p$ is measurement noise, $\boldsymbol{A}$, $\boldsymbol{B}$, $\boldsymbol{C}$ and $\boldsymbol{E}$ 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 $\boldsymbol{u}$ and the measurement $\boldsymbol{y}$ 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 $\tilde{\boldsymbol{u}} = \boldsymbol{u}$ and $\tilde{\boldsymbol{y}} = \boldsymbol{y}$ in Fig. 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 [14]. 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 $\boldsymbol{y}(k)$ using λ, a safe set can be defined for each system as follows:

$$\mathcal{S}_x = \left\{ \mathbf{x} : \max_k \{ \| \boldsymbol{C}\boldsymbol{x}(k) + \boldsymbol{v}(k) \|_\infty \} \leq \lambda \right\} \tag{2}$$

The system is said to be safe if the state trajectory $\boldsymbol{x}(k)$ remains in S_x. Therefore, the attacker, to damage the system, tries to drive the state of the system out of its safe set.

B. 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 [38].

1) PLANT

In our testbed, we use a ball and beam process as the plant, as shown in Fig. 2. A ball and beam system 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 set-point on top of the beam by tilting the beam using an electrical motor. We define a safe set for the ball and beam system by considering 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 $[-0.55\ m, 0.55\ m]$, 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 set-point but holds it on the beam, it may not damage the system but can cause extra cost and decrease efficiency. Hence, our aim in this paper is to hold the ball on the beam and the exact set-point.

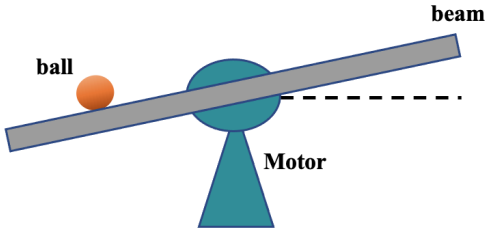


FIGURE 2. Ball and beam system.

We have chosen this system as a plant because it has a fast dynamic and is time critical, and even in the absence of attacks, 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 ($\boldsymbol{y}(t) = [y_1(t) \quad y_2(t) \quad y_3(t)]$): 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:

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

where $g = 9.80665$ is the gravity of Earth. We discretize this continuous time model with a sampling time of 0.05 s for designing the controller and our security framework. Also, we consider a sampling period of sensor measurements equal to 0.05 s. So, by discretizing this system, measurement signals $\boldsymbol{y_1}$, $\boldsymbol{y_2}$, and $\boldsymbol{y_3}$ will be discrete signals. $\boldsymbol{y_1}$ contains the positions of the ball on the beam at each sampling time, $\boldsymbol{y_2}$ contains the ball's speeds at each sampling time, and $\boldsymbol{y_3}$ contains the beam's angles at each sampling time. The allowed range for $\boldsymbol{y_1}$ is between $-0.55\ m$ and $0.55\ m$ regarding the beam length that is $1.1\ m$. Control signal $\boldsymbol{u}$ is also a discrete signal containing the control value generated by the controller. The control value is generated by the controller to determine the speed that should be set for the beam to adjust the ball's position on the beam.

2) CONTROLLER

We design an 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:

$$\begin{aligned} \underset{\mathbf{u}}{\text{minimize}}\ J = \sum_{i=k}^{k+N-1} \Big(& \boldsymbol{x}^T(i)\boldsymbol{Q}\boldsymbol{x}(i) + \boldsymbol{u}^T(i)\boldsymbol{R}\boldsymbol{u}(i) \\ & + \boldsymbol{x}^T(i+N)\boldsymbol{P}\boldsymbol{x}(i+N) \Big) \end{aligned}$$

$$\text{subject to } \boldsymbol{x}_{i+1} = \boldsymbol{A}\boldsymbol{x}_i + \boldsymbol{B}\boldsymbol{u}_i,$$
$$\boldsymbol{J}\begin{bmatrix}\boldsymbol{x}(i)\\ \boldsymbol{u}(i)\end{bmatrix} \leq \boldsymbol{j}, \quad \boldsymbol{H}\begin{bmatrix}\boldsymbol{x}(i)\\ \boldsymbol{u}(i)\end{bmatrix} = \boldsymbol{h},$$
$$\boldsymbol{x}(i+N) \in \mathcal{T} \tag{4}$$

where $\boldsymbol{Q}, \boldsymbol{R}$ and $\boldsymbol{P}$ are cost matrices, $\boldsymbol{A}$ and $\boldsymbol{B}$ define the model of the system, $\boldsymbol{x}$ is the state vector, $\boldsymbol{u}$ is the control signal, and the constraints of the system are defined by the matrices and vectors $\boldsymbol{J}, \boldsymbol{j}, \boldsymbol{H}$ and $\boldsymbol{h}$. $\mathcal{T}$ is called the terminal set and forces the final state to ensure the controller's stability. We deployed this controller in a Kubernetes cluster that will be described in the following.

3) 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 [39]. The cluster has been equipped with an Nginx ingress [40] and Prometheus operator [41]. 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.

C. 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 [42]. 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 $\boldsymbol{y}$ or control signal $\boldsymbol{u}$ in Fig. 1. In our previous work [43], we considered deception attacks on control signals $\boldsymbol{u}$, and we designed a method for detecting and mitigating it. Hence, in this paper, we consider deception attacks on measurement signals $\boldsymbol{y}$.

Assumption: We consider there is no attack on control signal $\boldsymbol{u}$, and we only have deception attacks on the measurement signal $\boldsymbol{y}$. However, we consider that it is possible to have an attack on several sensor measurements at the same time. We will examine our security framework for all $2^p - 2$ conditions for an attack occurring on $\boldsymbol{y}$, where p is the number of measurement signals: $\boldsymbol{y} \in \mathbb{R}^p$. 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 occurring an attack on the system.

The attacker adds an attack vector $\boldsymbol{f_a}(k) = [a_1(k)\ a_2(k) \dots a_p]^T$ to the measurement signal $\boldsymbol{y}(k) = [y_1(k) \quad y_2(k) \dots$ $y_p(k)]$, 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:

$$\tilde{\boldsymbol{y}}_k = \quad \boldsymbol{C}\boldsymbol{x}(k) + \boldsymbol{v}(k) + \boldsymbol{f_a}(k) \tag{5}$$

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.

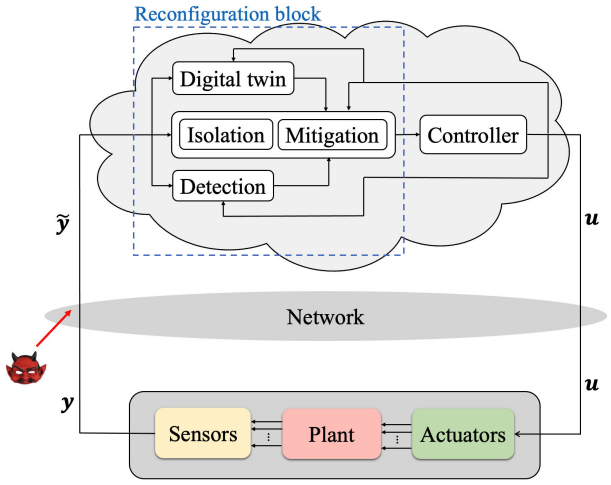


FIGURE 3. Proposed attack resilient framework overview.

IV. PROPOSED SOLUTION

In this section, we propose a security framework for cloud control systems to ensure the stability of the plant and maintain good 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. Fig. 3 demonstrates an overview of our proposed security framework. As shown in this figure, this framework includes attack detection, isolation, and mitigation parts, all deployed inside the cloud. Hence it is adaptable to the already implemented CCSs' frameworks, and we do not need to change them a lot. In the following, we will explain each part of our framework separately.

A. ATTACK DETECTION

In the attack detection part of our proposed security framework, we try to generate a residual signal such that it is close to zero and less than a predefined threshold in normal conditions 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).

1) 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 the observability of the system. Hence, by assuming that our system is observable, we propose designing a Kalman filter as an observer for the system based on our previous work [44]. As you can see in Fig. 4, the Kalman filter, by using control signals $\boldsymbol{u}$ and measurement signals $\boldsymbol{y}$, tries to estimate the correct value of the sensor measurements $\hat{\boldsymbol{y}}$.

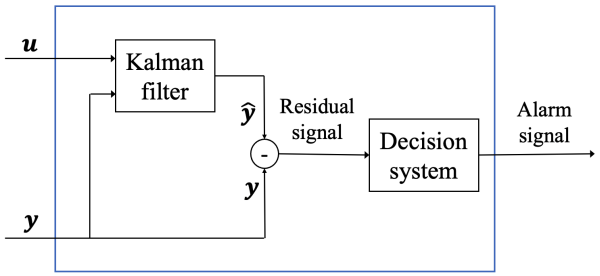


FIGURE 4. Observer-based attack detection overview.

For designing the Kalman filter, the measurement noise, which is added to the measurement signals, and the process noise, which describes the amount of uncertainty or deviation of the model from the real system, should be considered. By considering these noises in the system, it generally can be modeled as follows:

$$\begin{aligned} \boldsymbol{x}(k+1) &= \boldsymbol{A}\boldsymbol{x}(k) + \boldsymbol{B}\boldsymbol{u}(k) + \boldsymbol{G}\boldsymbol{w}(k) \quad \boldsymbol{w} \to \boldsymbol{N}(\boldsymbol{0}, \boldsymbol{Q}) \\ \boldsymbol{y}(k) &= \boldsymbol{C}\boldsymbol{x}(k) + \boldsymbol{F}\boldsymbol{v}(k) \quad \boldsymbol{v} \to \boldsymbol{N}(\boldsymbol{0}, \boldsymbol{R}) \end{aligned} \tag{6}$$

where $\boldsymbol{x}$ is the state vector, $\boldsymbol{y}$ is the measurement signal, $\boldsymbol{u}$ is the control signal, $\boldsymbol{w}$ is process noise, $\boldsymbol{v}$ is measurement noise and k shows time instance. Here, we consider process noise and measurement noise to be white noise with covariances $\boldsymbol{Q}$ and $\boldsymbol{R}$, respectively (here $\boldsymbol{Q}$ and $\boldsymbol{R}$ are not the same $\boldsymbol{Q}$ and $\boldsymbol{R}$ in (4)). Also, $\boldsymbol{A}$, $\boldsymbol{B}$, $\boldsymbol{C}$, $\boldsymbol{G}$, and $\boldsymbol{F}$ 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 [45]. The time update part consists of the following steps:

$$1) \quad \boldsymbol{x}(k|k-1) = \boldsymbol{A}(\mathrm{k})\hat{\boldsymbol{x}}(k-1|k-1) + \boldsymbol{B}(\mathrm{k})\boldsymbol{u}(k) \tag{7}$$

$$\begin{aligned} 2) \quad \boldsymbol{P}(k|k-1) &= \boldsymbol{G}(\mathrm{k}-1)\boldsymbol{Q}(\mathrm{k}-1)\boldsymbol{G}^{\mathrm{T}}(\mathrm{k}-1) \\ &\quad + \boldsymbol{A}(\mathrm{k}-1)\boldsymbol{P}(\mathrm{k}-1|\mathrm{k}-1)\boldsymbol{A}^{\mathrm{T}}(\mathrm{k}-1) \end{aligned} \tag{8}$$

and the measurement update part consists of the following steps:

$$\begin{aligned} 3) \quad \boldsymbol{K}(\mathrm{k}) &= \boldsymbol{P}(\mathrm{k}|\mathrm{k}-1)\boldsymbol{C}^{\mathrm{T}}(\mathrm{k})\big(\boldsymbol{C}(\mathrm{k})\boldsymbol{P}(\mathrm{k}|\mathrm{k}-1)\boldsymbol{C}^{\mathrm{T}}(\mathrm{k}) \\ &\quad + \boldsymbol{F}(\mathrm{k})\boldsymbol{R}(\mathrm{k})\boldsymbol{F}^{\mathrm{T}}(\mathrm{k})\big)^{-1} \end{aligned} \tag{9}$$

$$4) \quad \hat{\boldsymbol{x}}(k|k) = \hat{\boldsymbol{x}}(k|k-1) + \boldsymbol{K}\big(\mathrm{k}\big)\big(\boldsymbol{y}(\mathrm{k}) - \boldsymbol{C}(\mathrm{k})\hat{\boldsymbol{x}}(k|k-1)\big) \tag{10}$$

$$5) \quad \boldsymbol{P}(\mathrm{k}|\mathrm{k}) = \big(\boldsymbol{I} - \boldsymbol{K}(\mathrm{k})\boldsymbol{C}(\mathrm{k})\big)\boldsymbol{P}(\mathrm{k}|\mathrm{k}-1) \tag{11}$$

where $\hat{x}$ is the estimated state vector, P is the estimating covariance matrix and K is the Kalman gain. In a time-invariant system like (6), $A(k) = A(k-1) = A$ and the same rule is valid for other coefficient matrices in (6).

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

$$\hat{\boldsymbol{y}}(k) = C\hat{\boldsymbol{x}}(k) \tag{12}$$

Then by comparing $\boldsymbol{y}$ and $\hat{\boldsymbol{y}}$, residual signals can be generated as follows:

$$\boldsymbol{r}(k) = \boldsymbol{y}(k) - \hat{\boldsymbol{y}}(k) \tag{13}$$

where $\boldsymbol{r} \in \mathbb{R}^p$ that means for each measurement signal, there is a discrete residual signal. In normal conditions, the residual signal should be equal to zero, but the measurement noise causes some deviation from zero. Hence, we need a decision function for the evaluation of the residual signal, and it will determine whether an attack is present. For this, we use a decision function consisting of a test function and a threshold function based on our previous work [36], [44]. Test function $\varphi(r_p(k))$ provides a measure of the residual's deviation from zero as follows:

$$\varphi(r_p(k)) = |r_p(k)| \tag{14}$$

where r_p is the pth of residual signal related to pth of measurement signal. Then, the test function will be evaluated by a threshold function $\Phi(k)$ as follows:

$$\begin{cases} \mathcal{H}_0 : \varphi(r_p(k)) \leq \Phi_p(k) \\ \mathcal{H}_1 : \varphi(r_p(k)) > \Phi_p(k) \end{cases} \tag{15}$$

where hypothesis H_0 indicates normal operation of the system and H_1 indicates the abnormal mode of the system that triggers an alarm signal, and this should be checked for all residual signals related to all measurement signals.

In this paper, we suggest two different algorithms to determine the thresholds for attack detection. The first method that we have also used in our previous work [25] is Based on the 68-95-99.7 rule that says 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 noises in this paper are assumed Gaussian noise with zero mean, by considering a threshold equal to 3σ that σ is the standard deviation of measurement noise, 99.73% false alarms that may occur due to these noises can be filtered. The second method for determining threshold is based on [18] in that the author uses a set of healthy data, during which there are no attacks, for calculating appropriate thresholds, for example, the maximum value of the difference between data and their set-points can be used as the threshold. In this paper, we use this method to determine thresholds for residual signals and consider the absolute value of the maximum deviation of the residual signal from zero in normal conditions during which there is no attack.

2) ANALYTICAL REDUNDANCY RELATIONS

As is said in Section IV-A1, 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 uses measured variables and control signals as input. The main argument in the ARR method is that there is no need to use an 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:

$$\begin{aligned}\dot{\boldsymbol{x}}(t) &= \boldsymbol{g}(\boldsymbol{x}(t), \boldsymbol{u}(t), \boldsymbol{d}(t))\\ \boldsymbol{y}(t) &= \boldsymbol{h}(\boldsymbol{x}(t), \boldsymbol{u}(t), \boldsymbol{d}(t))\end{aligned} \tag{16}$$

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

$$\begin{aligned}\mathcal{H}_0 \Leftrightarrow\ & [\dot{\boldsymbol{x}}(t) = \boldsymbol{g}(\boldsymbol{x}(t), \boldsymbol{u}(t), \boldsymbol{d}(t))]\\ & \wedge [\boldsymbol{y}(t) = \boldsymbol{h}(\boldsymbol{x}(t), \boldsymbol{u}(t), \boldsymbol{d}(t))]\end{aligned} \tag{17}$$

$$\begin{aligned}\mathcal{H}_1 \Leftrightarrow\ & [\dot{\boldsymbol{x}}(t) \neq \boldsymbol{g}(\boldsymbol{x}(t), \boldsymbol{u}(t), \boldsymbol{d}(t))]\\ & \vee [\boldsymbol{y}(t) \neq \boldsymbol{h}(\boldsymbol{x}(t), \boldsymbol{u}(t), \boldsymbol{d}(t))]\end{aligned} \tag{18}$$

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

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

$$(q+1)p > n + (q+1)n_d \tag{19}$$

Regarding $\boldsymbol{y} \in \mathbb{R}^p$, 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 $\boldsymbol{x} \in \mathbb{R}^n$, disturbance and its differentiation $\bar{\boldsymbol{d}}^{(q)} \in \mathbb{R}^{(q+1)n_d}$. In this paper, $z^{(q)}$ indicate the qth order derivative of variable z, and we have $\bar{\boldsymbol{z}}^{(q)} = [z \quad \dot{z} \ldots z^{(q)}]^T$. Thus, to have enough 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 (19), 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 ARRs.

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

$$\begin{aligned}\boldsymbol{r}\left(\bar{\boldsymbol{y}}^{(q)}, \bar{\boldsymbol{u}}^{(q)}\right) &= \boldsymbol{0} \Leftrightarrow \mathcal{H}_0\\ \boldsymbol{r}\left(\bar{\boldsymbol{y}}^{(q)}, \bar{\boldsymbol{u}}^{(q)}\right) &\neq \boldsymbol{0} \Leftrightarrow \mathcal{H}_1\end{aligned} \tag{20}$$

Algorithm 1 Algorithm for Finding ARRs

Input: n, p, n_d and system's model by considering attack vector $\boldsymbol{f_a}$:

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

Output: ARRs

1: Find the minimum q that satisfies:

$$(q+1)p > n + (q+1)n_d$$

2: Find matrix $\boldsymbol{H^q}$ that includes output equations and their differentiation up to qth order of derivative:

$$\begin{bmatrix}\boldsymbol{y}\\ \dot{\boldsymbol{y}}\\ \ddot{\boldsymbol{y}}\\ \vdots\\ \boldsymbol{y}^{(q)}\end{bmatrix} = \begin{bmatrix}\boldsymbol{h}(\boldsymbol{x}, \boldsymbol{u}, \boldsymbol{d}, \boldsymbol{f_a})\\ \boldsymbol{h_1}\left(\boldsymbol{x}, \bar{\boldsymbol{u}}^{(1)}, \bar{\boldsymbol{d}}^{(1)}, \bar{\boldsymbol{f_a}}^{(1)}\right)\\ \boldsymbol{h_2}\left(\boldsymbol{x}, \bar{\boldsymbol{u}}^{(2)}, \bar{\boldsymbol{d}}^{(2)}, \bar{\boldsymbol{f_a}}^{(2)}\right)\\ \vdots\\ \boldsymbol{h_q}\left(\boldsymbol{x}, \bar{\boldsymbol{u}}^{(q)}, \bar{\boldsymbol{d}}^{(q)}, \bar{\boldsymbol{f_a}}^{(q)}\right)\end{bmatrix} = \boldsymbol{H^q}$$

3: **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**

q = q + 1;

Find the new $\boldsymbol{H^q}$ based on step 2 but using the new q

4: **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**

Use at least the $n + (q+1)n_d$ first equations in $\boldsymbol{H^q}$ to find unknown variables $\boldsymbol{x}$ and $\bar{\boldsymbol{d}}^{(q)} = [d \quad d^{(1)} \quad \ldots \quad d^{(q)}]$ based on known variables:

$$\begin{bmatrix}\boldsymbol{x}\\ \bar{\boldsymbol{d}}^{(q)}\end{bmatrix} = \begin{bmatrix}\boldsymbol{\phi_x}\left(\bar{\boldsymbol{y}}_M^{(q)}, \bar{\boldsymbol{u}}^{(q)}, \bar{\boldsymbol{f_a}}^{(q)}\right)\\ \boldsymbol{\phi_d}\left(\bar{\boldsymbol{y}}_M^{(q)}, \bar{\boldsymbol{u}}^{(q)}, \bar{\boldsymbol{f_a}}^{(q)}\right)\end{bmatrix}$$

Substitute these obtained variables in the remained equations of $\boldsymbol{H^q}$ and put $\boldsymbol{f_a}(t) = 0$ to find ARRs:

$$\boldsymbol{0} = \boldsymbol{r}\left(\bar{\boldsymbol{y}}^{(q)}, \bar{\boldsymbol{u}}^{(q)}, \boldsymbol{0}\right)$$

Using algorithm 1, we can find ARRs for our testbed described in Section III-B as shown below:

$$\boldsymbol{r}(t) = \begin{bmatrix}r_1(t)\\ r_2(t)\\ r_3(t)\end{bmatrix} = \begin{bmatrix}\dot{y}_1(t) - \frac{a_1}{b_1}y_2(t)\\ \dot{y}_2(t) + \frac{b_2}{c_3}\frac{5g}{7}y_3(t)\\ \dot{y}_3(t) - 0.44\, cu(t)\end{bmatrix} \tag{21}$$

As can be seen, these residual signals are composed of only the outputs $\boldsymbol{y}$, the outputs' derivatives $\boldsymbol{y}^{(q)}$, and the input $\boldsymbol{u}$. In normal conditions, 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.

B. 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.

1) ANALYTICAL REDUNDANCY RELATIONS

In Section IV-A2, it was explained how ARRs 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 (20), obtained residuals from the ARR method are only dependent on measurement signals and its derivatives $\bar{\boldsymbol{y}}^{(q)}$ as well as control signals and its derivatives $\bar{\boldsymbol{u}}^{(q)}$. In the 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 III-B, based on the residuals that we found for it in (21), 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 1:

TABLE 1. 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

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 the signature for the attack on y_2. In Table 1, we can see each attack has a unique signature that makes us able to diagnose on which measurement signal the attack has occurred.

Although the ARRs isolation method seems simple, it is completely dependent on the model of the system and also how ARRs are related to the measurement signals. Hence, we cannot guarantee that always works. Also, as said before, in this paper, we consider that it is possible to have simultaneous attacks on several measurement signals. consequently, we need an isolation method that can be used for the isolation of such simultaneous attacks. However, the ARR method cannot guarantee that. For example, in our testbed case, if there are two simultaneous attacks on y_1, and y_2, based on Table 1, 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 a simultaneous attack on y_1 and y_2, we have changes in r_1 and r_2 and the signature for this attack will be $(011)_2 = 3$ that is the same as the signature of the single attack on y_2.Therefore, regarding these defects of the ARR isolation method, in the next section, we propose a novel isolation method that is able to isolate both single and simultaneous attacks. All other defects of the ARR isolation method are discussed in Section VI.

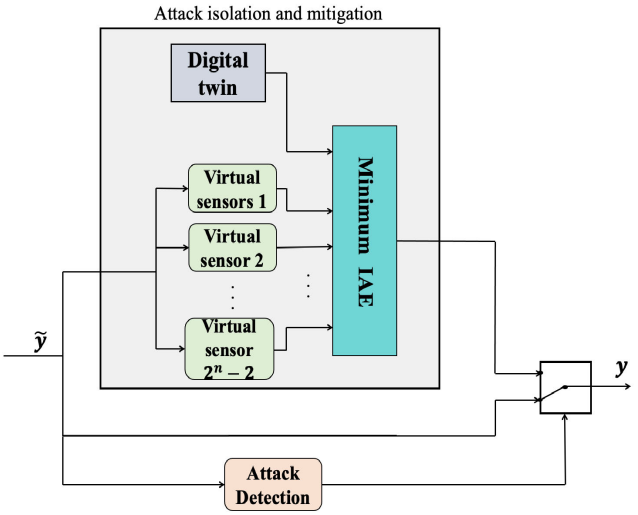


FIGURE 5. Digital twin-based attack isolation and mitigation.

2) PROPOSED DIGITAL TWIN-BASED ISOLATION METHOD

We need isolation because based on our attack mitigation method that will be explained in Section IV-C, after detecting the attack, 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 the digital twin 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.

Fig.5 shows the virtual sensors that each of them has designed for each mode. They use the measurement and control signals, and depending on the fact that each virtual sensor works in which mode, it assumes one or some measurement signals are under attack and removes them and uses 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 and one of the most important Industry 4.0 technologies. With

digital twins, we have virtual replicas of physical systems so that they precisely mirror the internal behavior of the physical systems [46]. Hence, in our isolation method, we will take advantage of digital twins to define a reference operation that we can use for comparing the 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 to define this difference, we integrate the absolute error (IAE) as follows:

$$E_j = \sum_{i=1}^{k_n} \left| z_i - z_{DT_i} \right|, \quad j \in \{1, 2, \ldots, 2^n - 2\} \tag{22}$$

where k_n is a window that we use to calculate 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 the output of virtual sensors (z) with the output of the digital twin (z_{DT}) and this will lead to having a more reliable choice than when only calculate the difference at current step k. The size of this window time depends on the dynamic of the plant. For example, the ball and beam system has a fast dynamic and reacts to changes immediately. So, considering a small window can cover its divergence from the normal condition. However, for a plant with a slow dynamic, like the quadruple tank process, a large window time should be chosen since it reacts to changes gradually.

In (22), z is the output of a virtual sensor, and z_{DT} is the output of the digital twin. So, in each time instant k, we calculate $2^n - 2$ errors (IAE). When the alarm signal from the detection part shows there is an attack in the system, we choose the virtual sensor that has the minimum error between these $2^n - 2$ calculated errors 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 the isolation part, the mathematical model of the plant is used for creating the digital twin based on [44] and [47].

C. ATTACK MITIGATION

In our previous work [36], we proposed to employ an ancillary controller in the physical domain to mitigate the impacts of the attack such that once the attack has been detected, we switch from the 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 the already implemented CCSs 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 systems effortlessly.

In our mitigation method, we try to hide the attack from the controller, and as you can see in Fig. 6, we add a reconfiguration block in the cloud close to the controller, and it gets the measurement signal that has been manipulated by

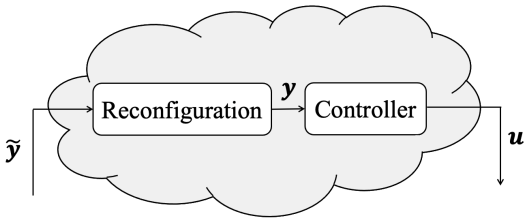


FIGURE 6. Hiding the attack from the controller using reconfiguration block.

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 the controller, and the controller will generate the correct control signal based on the output of the reconfiguration block.

In the reconfiguration block in Fig. 6, to reconstruct the measurement signal, we utilize the virtual sensor that is also explained in the isolation part. In the model of the plant (1), each row of matrix $\boldsymbol{C}$ is related to each sensor measurement. Based on the isolation part, we can diagnose that the attack has occurred on which sensors, and by removing the rows relates to the sensors that we have an attack on, we can generate matrix $\boldsymbol{C_a}$. Using this, the system with attacks can be described by the state-space model:

$$\begin{aligned} \dot{\boldsymbol{x}}_{\boldsymbol{a}}(t) &= \boldsymbol{A}\boldsymbol{x}_{\boldsymbol{a}}(t) + \boldsymbol{B}\boldsymbol{u}(t) + \boldsymbol{E}\boldsymbol{d}(t) \\ \boldsymbol{y}_{\boldsymbol{a}}(t) &= \boldsymbol{C}_{\boldsymbol{a}}\boldsymbol{x}_{\boldsymbol{a}}(t) \end{aligned} \tag{23}$$

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

$$\text{rank} \begin{bmatrix} \boldsymbol{C_a} \\ \boldsymbol{C_a A} \\ \boldsymbol{C_a A}^2 \\ \vdots \\ \boldsymbol{C_a A}^{n-1} \end{bmatrix} = n \tag{24}$$

where n is the dimension of the matrix $\boldsymbol{A} \in R^{n \times n}$. If condition (24) is satisfied, it means $(\boldsymbol{A}, \boldsymbol{C_a})$ is observable, and the entire state vector can be reconstructed. Regarding this condition, we will consider the following assumption.

Assumption: By checking the 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 we need to meet observability conditions for all attack modes as protected measurements such that attackers cannot access them.

TABLE 2. Different modes of attack.

	Attack on y_1	Attack on y_2	Attack on y_3
mode 1	-	-	-
mode 2	×	-	-
mode3	-	×	-
mode 4	-	-	×
mode 5	×	×	-
mode 6	×	-	×
mode 7	-	×	×

Now we can design a virtual sensor based on [17] as follows:

$$\begin{aligned} \dot{\boldsymbol{x}}_{\boldsymbol{V}}(t) &= \boldsymbol{A}_{\boldsymbol{V}}\boldsymbol{x}_{\boldsymbol{V}}(t) + \boldsymbol{B}_{\boldsymbol{V}}\boldsymbol{u}_{\boldsymbol{c}}(t) + \boldsymbol{L}\boldsymbol{y}_{\boldsymbol{a}}(t) \\ \boldsymbol{y}_{\boldsymbol{c}}(t) &= \boldsymbol{C}_{\boldsymbol{V}}\boldsymbol{x}_{\boldsymbol{V}}(t) + \boldsymbol{P}\boldsymbol{y}_{\boldsymbol{a}}(t) \end{aligned} \tag{25}$$

that we have:

$$\begin{aligned} \boldsymbol{A}_{\boldsymbol{V}} &= \boldsymbol{A} - \boldsymbol{L}\boldsymbol{C}_{\boldsymbol{a}} \\ \boldsymbol{B}_{\boldsymbol{V}} &= \boldsymbol{B} \\ \boldsymbol{C}_{\boldsymbol{V}} &= \boldsymbol{C} - \boldsymbol{P}\boldsymbol{C}_{\boldsymbol{a}} \\ \boldsymbol{P} &= \boldsymbol{C}\boldsymbol{C}_{\boldsymbol{a}}{}^{+} \end{aligned} \tag{26}$$

in (26), $\boldsymbol{L}$ is chosen such that $\boldsymbol{A} - \boldsymbol{L}\boldsymbol{C}_{\boldsymbol{a}}$ is Hurwitz.

For designing a discrete-time virtual sensor, equivalent equations can be used by substituting matrices with discrete-time system model matrices.

V. EXPERIMENTS

To evaluate our proposed security framework, we deploy it on the real testbed that was explained in Section III-B, and in this section, we describe our experiments. In our testbed, we have three sensors for measuring the position of the ball, the speed of the ball, and the angle of the beam. Hence, we can have $2^3 = 8$ different combinations of these measurement signals and by disregarding the case in which we have attacks on all measurement signals (based on our assumption in Section III-C it is not possible), we can define $2^3 - 1 = 7$ different modes as shown in Table 2. In this table, mode 1 is related to the 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, $\boldsymbol{f_a}(k) = [a_1(k) \quad a_2(k) \quad a_3(k)]^T$ is added to measurements signals and depending on that we have attack(s) on which signal(s), a_i can be zero or non zero. In our experiments, we generate these non-zero values as follows:

$$a_i = \mathcal{N}\left(\mu, \sigma^2\right), \quad \mu = \lambda_r(k_j - k_0) \tag{27}$$

where σ^2 is constant and μ increases with time. In fact, by applying this attack, we will increase the real value of the measurement signal gradually with time, such that detecting the attack becomes 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 conditions, we utilize Chaos Mesh [48]. 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 for applying different amounts of delay in the Round-Trip Time (RTT) between the plant and the controller in the cloud to create the real condition in which there may be different amounts of network delays.

A. EVALUATION OF ATTACK DETECTION METHODS

In the first part of the experiment, we evaluate the attack detection part. In (27), 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 (27) is reasonable since it will cause the ball to fall off, and also, it won't be easy 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 IV-A, for each mode in Table 2, we generate attacks based on (27) 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 the efficiency of these two different methods for detecting attacks by measuring the time it takes to detect the attack.

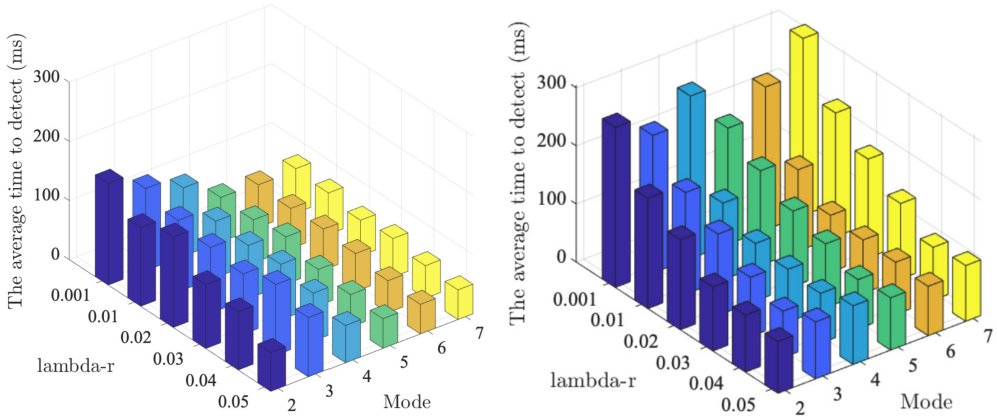
B. EVALUATION OF ATTACK ISOLATION METHODS

In Section IV-B, we provided two different isolation methods for determining the location of the attack. In the second part of our experiment, we evaluate these isolation methods 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 the ARR method and the effectiveness of our proposed isolation method.

C. EVALUATION OF ATTACK MITIGATION METHOD

In the third part of our experiment, we evaluate our mitigation method. For this, we apply different modes of attack on our testbed and then we investigate how we can mitigate the attack. For this part, we detect attacks using observer-based attack detection and isolate them using our proposed isolation method. Also, as a 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.

$$IAE = \sum_{k=0}^{T} |y(k) - s(k)|, \tag{28}$$



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

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

FIGURE 7. The average time to detect attacks using observer-based and ARR attack detection methods.

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 a performance metric for evaluating our mitigation method.

VI. RESULTS AND DISCUSSION

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

A. ATTACK DETECTION

Regarding Section V-A, for evaluating the attack detection part, for each mode of Table 2 except mode 1 that shows the 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, which equals 750 s, and apply the attack on the 14500th sample, that is 725 s, and then we measure how much time it takes to detect this attack. For each mode and each λ, by applying different delays 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. Fig. 7 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, the 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 the observer-based method, and it is 305 ms using the ARR method. Hence, for other attacks with larger λ_r, it takes less than this time to detect the attack, which means both of these attack detection methods can detect attacks fast. By comparing Fig. 7a and Fig. 7b, it can be seen the time to detect attacks using both methods is close.

TABLE 3. ARR signatures for each mode of attack.

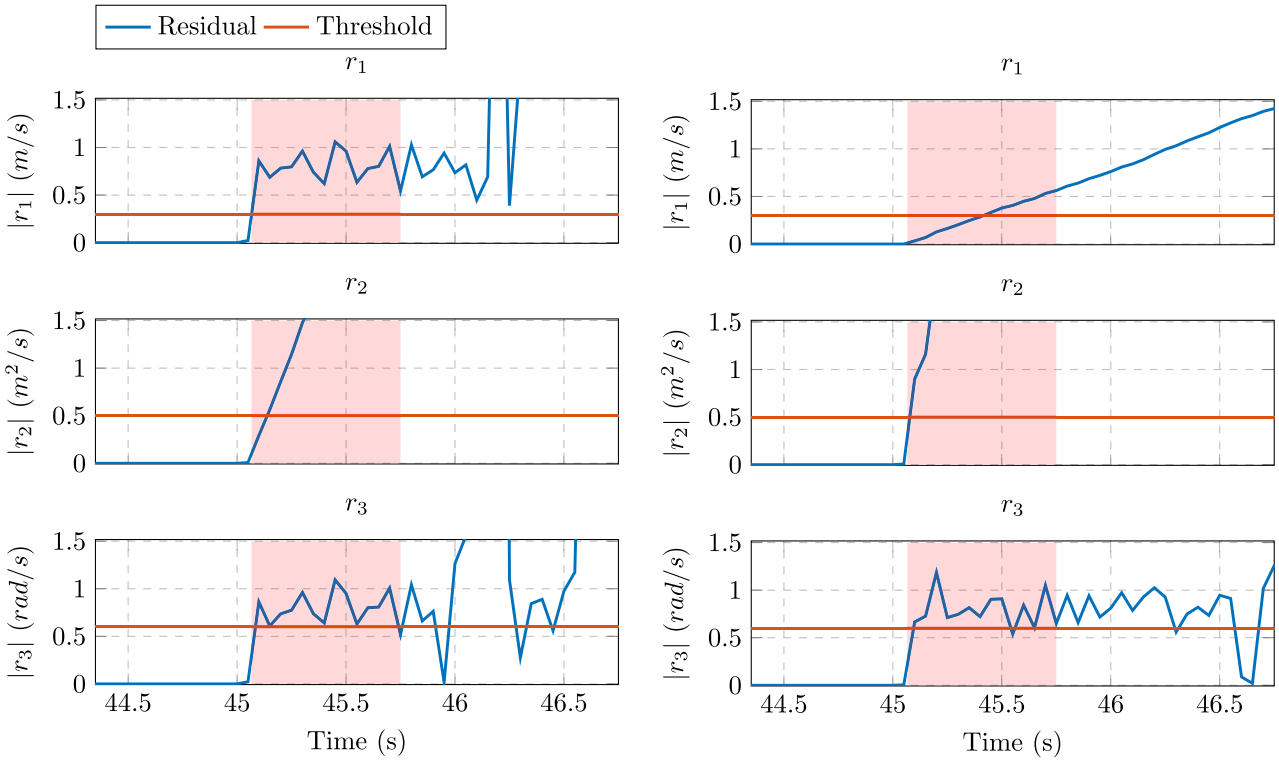
Modes	Mode 2	Mode 3	Mode 4	Mode 5	Mode 6	Mode 7
Signature	1	3	6	3	7	7

However, on average, the time to detect the attack in each mode using observer-based attack detection is shorter than the ARR attack detection method.

B. ATTACK ISOLATION

In the ARR-based isolation method, based on Table 1, we define the signatures for each mode of attack in Table 2 for our testbed in Table 3. As can be seen in this table, modes 3 and 5, and also modes 6 and 7, have the same signature. So, when we get the signature of 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 the signature as 3, we are not able to diagnose that the attack has occurred on both the position signal and speed signal or only on the speed signal. Fig. 8 shows residual signals for the ARR-based isolation method for modes 6 and 7. In this figure, all attacks are applied on the system at 45 s. Fig. 8a is related to mode 6, and in this mode, we have simultaneous attacks on the position signal and angle signal and based on Table 1, this attack will have an effect on r_1 due to the attack on position signal (y_1) and also it will have an effect on r_2 and r_3 due to the attack on angle signal (y_3). So, as can be seen in Fig. 8a as we expect all residuals exceed their threshold and as the signature for this attack, we will have $(r_1 r_2 r_3) = (111)_2 = 7$.

Fig. 8b is related to mode 7, and in this mode, we have simultaneous attacks on the speed signal and angle signal. based on Table 1, 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 Fig. 8b as we expect all residuals exceed their threshold and as the signature for this attack, we will have



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

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

FIGURE 8. Residual signals for the ARR-based isolation method for mode 6 and 7.

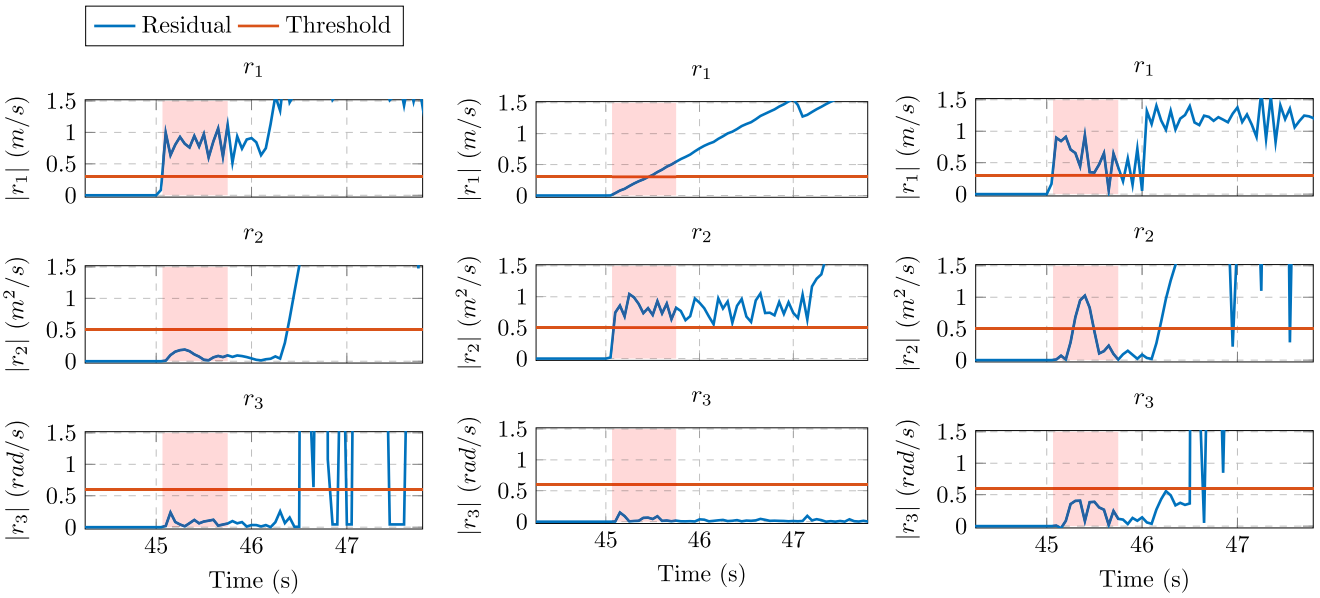
$(r_1 r_2 r_3) = (111)_2 = 7$. So, for both cases, modes 6 and 7, we got the same signature 7, and that means that if we get signature 7, we will not be able to distinguish whether we are in mode 6 or 7.

Therefore, one of the main weak points of the ARR-based isolation method is that this method is completely dependent on the model of the plant, and based on that, signatures of different modes of attack will be defined, and we do not have any control over that. Hence, we may have similar signatures for different modes and not be able to diagnose the correct mode of attack, as we had this problem for our testbed in Table 3. However, we do not have such a 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 the ARR-based isolation method. The first one that has a 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 the signature; otherwise, it will be considered as zero. The second issue is related to the 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 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. The red area in Fig. 8 and 9 denotes the window time.

It is difficult to choose thresholds for residual signals and window time because it should work for all conditions. A smaller threshold will lead to the residual signal exceeding the threshold, and we will have faster isolation, but it will also cause some false alarms, and the residuals that should be considered zero will be considered as one, and consequently, we will have a wrong signature and wrong isolation. Longer window time is more conservative and causes not missing the residuals that will exceed their thresholds a bit later. However, the longer window time will lead to waiting longer, and consequently, it takes more time to do isolation, and this will affect 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.

Fig. 9a shows residual signals for the condition in which we have an attack on position signal at 45 s. So, in this condition based on Table 1, we expect only r_1 to exceed its threshold to create signature $(001)_2 = 1$. Regarding our chosen thresholds in Fig. 9a, 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 9. Threshold challenges in ARR_based isolation method.

r_1 exceeds its threshold as we expected. Fig. 9b also shows residual signals for the condition we have an attack on speed signal at 45 s. So, it seems chosen thresholds work well, and as we expect based on Table 1 and 3, during the window time, r_1 and r_2 exceed their threshold. However, in Fig. 9c residual signals for the condition, we have an attack on position signal at 45 s, 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 Fig. 9a, we expect only r_1 to exceed its threshold during window time, but we can see r_2 also has exceeded its threshold, which will lead to having a wrong signature and wrong isolation.

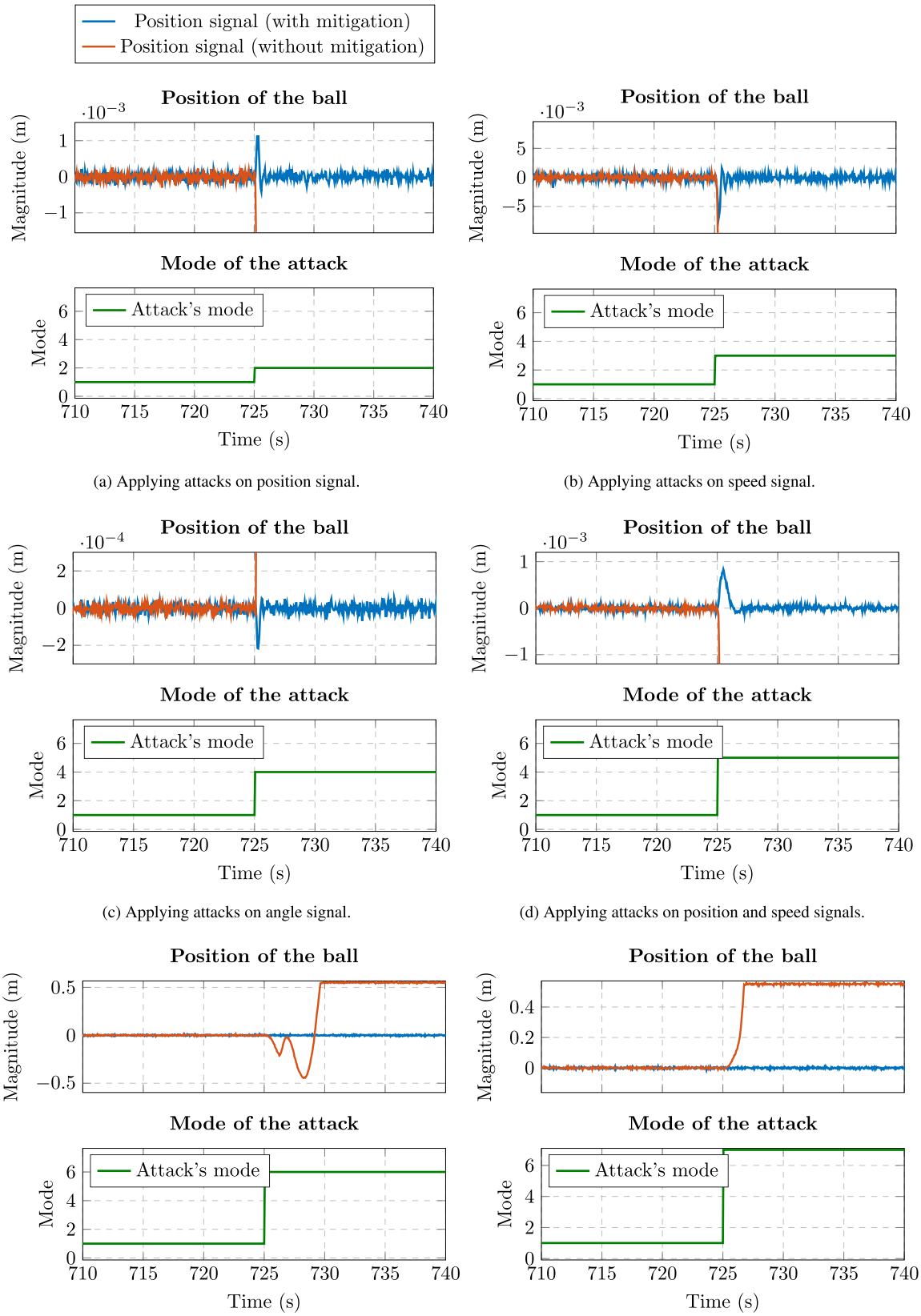
To solve such a problem, we can either increase the threshold for r_2 or decrease the window time, but both of these changes are so challenging. For example, here, if we want to increase the threshold for r_2 to solve the problem in Fig. 9c such that r_2 that has passed the threshold remains under the threshold, it will cause a problem in Fig. 9b since in this figure r_2 is supposed to surpass the threshold, but if we increase the threshold such that r_2 in Fig. 9c remains under the threshold, r_2 in Fig. 9b also will be so close to the threshold or lower than it that will cause misleading and generating the wrong signature.

On the other hand, decreasing the window time for solving the problem in Fig. 9c, 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 the signature, this may cause missing the residuals that will exceed their thresholds a bit later. For example, r_1 in Fig. 9b 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 waiting more time, and consequently, it takes more time to do isolation, and this will affect attack mitigation. For example, this can solve the problem of r_2 in Fig. 9c, but in Fig. 9b, we should wait for 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, the ARR-based isolation method not only does not work for attacks that cause the same signature but also choosing the appropriate threshold and window time is very challenging and has a 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 has very low efficiency as an isolation method. Because in attack detection using ARR, we care only about the first residual signal that exceeds its threshold, but all residual signals should be considered for isolation.

Fig. 10 shows our proposed isolation method's performance to determine which measurement signals the attack has occurred and specify the mode of attack. In this figure, we generate the attack based on (27) with $\lambda_r = 0.04$ and apply it to related measurement signals to each mode of Table 2 at 725 s. For instance, in mode 2, we apply this attack only to the position signal, and in mode 6, we apply it to both position and angle signals. Also, using Chaos Mesh, we apply a network delay of 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 Fig. 10, in all six modes



(a) Applying attacks on position signal.

(b) Applying attacks on speed signal.

(c) Applying attacks on angle signal.

(d) Applying attacks on position and speed signals.

(e) Applying attacks on position and angle signals.

(f) Applying attacks on speed and angle signals.

FIGURE 10. Isolation using our proposed method and mitigation based on this isolation.

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 precisely specify the mode of the attack and diagnose on which sensor there is an attack.

C. ATTACK MITIGATION

In the following, based on the detection and isolation from the previous section, we activate our mitigation part to mitigate the attack's impact. In Fig. 10, we can also see the position signal for each mode of attack. As seen in the figure, the blue curve shows the ball's position on the beam in the presence of our mitigation method, and the red curve shows the ball's position on the beam in the absence of the mitigation method. For example, in Fig. 10d, the attack applies to position and speed signals simultaneously at 725 s 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 after three sampling times at 725.15 s and activates our proposed isolation method. Then, our proposed isolation method specifies mode 5 for this attack based on Table 2 means that there are attacks on position and speed signal. Based on this isolation, in our mitigation part, in the reconfiguration block in Fig. 6, before feeding measurement signals to the controller, position, and speed signals are removed and regenerated using the rest of the measurement signals and then these new signals are fed to the controller. By doing this, as the blue curve in Fig. 10d 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 Fig. 10d, and at the end, it will fall off from the end of the beam.

Regarding Section V-C, to evaluate our mitigation method and to see if it can keep the system stable with good performance, we measure the controller's performance using IAE. Fig. 11 shows IAE for the normal condition during which there are 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 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 the normal condition, which proves that we can keep the plant stable with good performance during the attack.

VII. CONCLUSION

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-based methods were investigated and evaluated as attack detection in this framework. We showed both methods have acceptable performance and can detect attacks fast, but the observer-base method can detect attacks in a shorter time.

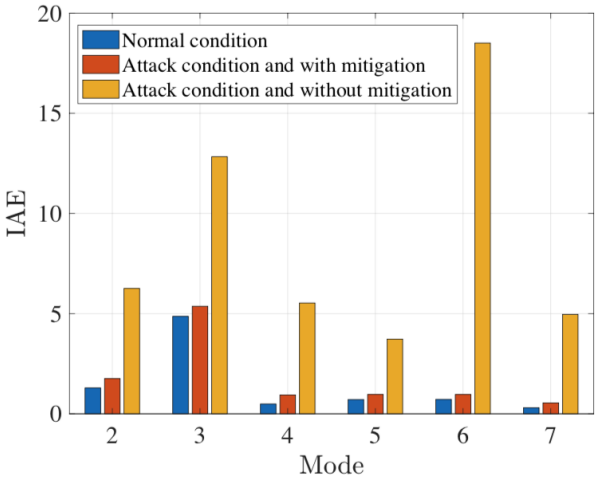


FIGURE 11. IAE for each mode of the system.

In the isolation part, first, we evaluated the available isolation method ARR and showed that the ARR-based isolation method not only does not work for attacks that cause the same signature but also choosing the appropriate threshold and window time is very challenging and has a significant 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 has very low efficiency as an isolation method. Regarding the defects of this method, we proposed a novel approach by combining the 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 can 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 on pure control theory, and gives a whole new way to approach these types of problems that ties into current hot research trends plus it works better than previous methods.

We also proposed a mitigation part in this framework by developing the virtual sensor concept for cloud control systems based on fault-tolerant control systems. By applying different modes of attack on the system, we proved that this mitigation method could keep the system stable with a good 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 tolerate this attack using our proposed framework.

Future work to investigate other kinds of attacks on CCSs, like Replay attacks, will be carried out to design some methods to detect and mitigate these kinds of attacks. Also, we will study some new techniques for delay compensation between the cloud and the plant. In this paper, we used an MPC controller for this objective, and using its predictive features,

we tried to deal with the delay problem. As a future work, we will design delay compensation methods that allow us to have even simpler controllers inside the cloud instead of MPC.

REFERENCES

[1] L. S. Dalenogare, G. B. Benitez, N. F. Ayala, and A. G. Frank, "The expected contribution of industry 4.0 technologies for industrial performance," *Int. J. Prod. Econ.*, vol. 204, pp. 383–394, Oct. 2018.

[2] K. Kumar, D. Zindani, and J. P. Davim, *Industry 4.0: Developments Towards the Fourth Industrial Revolution*. Cham, Switzerland: Springer, 2019.

[3] M. A. Khan and K. A. Abuhasel, "An evolutionary multi-hidden Markov model for intelligent threat sensing in industrial Internet of Things," *J. Supercomput.*, vol. 77, no. 6, pp. 6236–6250, Jun. 2021.

[4] M. A. Khan and K. A. Abuhasel, "Advanced metameric dimension framework for heterogeneous industrial Internet of Things," *Comput. Intell.*, vol. 37, no. 3, pp. 1367–1387, Aug. 2021.

[5] Y. Xia, "Cloud control systems," *IEEE/CAA J. Autom. Sinica*, vol. 2, no. 2, pp. 134–142, Apr. 2015.

[6] X. Xu, Y. Lu, B. Vogel-Heuser, and L. Wang, "Industry 4.0 and industry 5.0—Inception, conception and perception," *J. Manuf. Syst.*, vol. 61, pp. 530–535, Oct. 2021.

[7] M. A. Khan and N. S. Alghamdi, "A neutrosophic WPM-based machine learning model for device trust in industrial Internet of Things," *J. Ambient Intell. Hum. Comput.*, pp. 1–15, Aug. 2021, doi: 10.1007/s12652-021-03431-2.

[8] R. Waslo, T. Lewis, R. Hajj, and R. Carton, "Industry 4.0 and cybersecurity: Managing risk in an age of connected production," *Erişim Tarihi*, vol. 15, Mar. 2017. [Online]. Available: https://www2.deloitte.com/us/en/insights/focus/industry-4-0/cybersecurity-managing-risk-in-age-of-connected-production.html

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

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

[11] ICS-CERT. (Mar. 2017). *Hatman-Safety System Targeted Malware*. [Online]. Available: https://ics-cert.us-cert.gov/MAR-17-352-01-HatManTargeted-Malware

[12] Kaspersky Lab ICS-CERT. (Mar. 2018). *Threat Landscape for Industrial Automation Systems in H2 2017*. [Online]. Available: https://icscert.kaspersky.com/reports/2018/03/26/threat-landscape-for-industrial-automation-systems-in-h2-2017/

[13] N. S. Malik, R. Collins, and M. Vamburkar. (Apr. 2018). *Cyber-Attack, Pings Data Systems of at Least Four Gas Networks*. [Online]. Available: https://www.bloomberg.com/news/articles/2018-04-03/day-after-cyberattack-a-third-gas-pipeline-data-system-shuts

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

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

[16] R. Mattioli and K. Moulinos, "Communication network interdependencies in smart grids," in *EUA FNAI Security*. Athens, Greece: ENISA, 2015.

[17] M. Blanke, M. Kinnaert, J. Lunze, M. Staroswiecki, and J. Schröder, *Diagnosis and Fault-Tolerant Control*, vol. 2. Berlin, Germany: Springer, 2016.

[18] 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," *Proc. IEEE*, vol. 106, no. 1, pp. 113–128, Jan. 2018.

[19] Y. Yang, X. Yang, M. Heidari, M. A. Khan, G. Srivastava, M. Khosravi, and L. Qi, "ASTREAM: Data-stream-driven scalable anomaly detection with accuracy guarantee in IIoT environment," *IEEE Trans. Netw. Sci. Eng.*, early access, Mar. 8, 2022, doi: 10.1109/TNSE.2022.3157730.

[20] J. Zhang, L. Pan, Q.-L. Han, C. Chen, S. Wen, and Y. Xiang, "Deep learning based attack detection for cyber-physical system cybersecurity: A survey," *IEEE/CAA J. Autom. Sinica*, vol. 9, no. 3, pp. 377–391, Mar. 2022.

[21] L. Li, H. Yang, Y. Xia, and C. Zhu, "Attack detection and distributed filtering for state-saturated systems under deception attack," *IEEE Trans. Control Netw. Syst.*, vol. 8, no. 4, pp. 1918–1929, Dec. 2021.

[22] A. Naha, A. Teixeira, A. Ahlén, and S. Dey, "Deception attack detection using reduced watermarking," in *Proc. Eur. Control Conf. (ECC)*, Jun. 2021, pp. 74–80.

[23] I. Bessa, C. Trapiello, V. Puig, and R. M. Palhares, "Dual-rate control framework with safe watermarking against deception attacks," *IEEE Trans. Syst., Man, Cybern., Syst.*, vol. 52, no. 12, pp. 7494–7506, Dec. 2022.

[24] A. S. L. V. Tummala and R. K. Inapakurthi, "A two-stage Kalman filter for cyber-attack detection in automatic generation control system," *J. Mod. Power Syst. Clean Energy*, vol. 10, no. 1, pp. 50–59, 2022.

[25] F. Akbarian, A. Ramezani, M. Hamidi-Beheshti, and V. Haghighat, "Advanced algorithm to detect stealthy cyber attacks on automatic generation control in smart grid," *IET Cyber-Phys. Syst., Theory Appl.*, vol. 5, no. 4, pp. 351–358, Dec. 2020.

[26] J. Yang, W.-A. Zhang, and F. Guo, "Distributed Kalman-like filtering and bad data detection in the large-scale power system," *IEEE Trans. Ind. Informat.*, vol. 18, no. 8, pp. 5096–5104, Aug. 2021.

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

[28] F. van Wyk, Y. Wang, A. Khojandi, and N. Masoud, "Real-time sensor anomaly detection and identification in automated vehicles," *IEEE Trans. Intell. Transp. Syst.*, vol. 21, no. 3, pp. 1264–1276, Mar. 2020.

[29] Z. Ju, H. Zhang, and Y. Tan, "Distributed deception attack detection in platoon-based connected vehicle systems," *IEEE Trans. Veh. Technol.*, vol. 69, no. 5, pp. 4609–4620, May 2020.

[30] Y. Fang, H. Min, W. Wang, Z. Xu, and X. Zhao, "A fault detection and diagnosis system for autonomous vehicles based on hybrid approaches," *IEEE Sensors J.*, vol. 20, no. 16, pp. 9359–9371, Aug. 2020.

[31] 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 Trans. Ind. Informat.*, vol. 16, no. 6, pp. 4281–4292, Jun. 2020.

[32] C. Fang, Y. Qi, J. Chen, R. Tan, and W. X. Zheng, "Stealthy actuator signal attacks in stochastic control systems: Performance and limitations," *IEEE Trans. Autom. Control*, vol. 65, no. 9, pp. 3927–3934, Sep. 2020.

[33] W. E. Sayed, A. Aboelhassan, A. Hebala, G. Buticchi, and M. Galea, "Three tank system sensors and actuators faults detection employing unscented Kalman filter," in *Proc. IEEE 19th Int. Power Electron. Motion Control Conf. (PEMC)*, Apr. 2021, pp. 899–905.

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

[35] 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 Appl.*, vol. 12, no. 10, pp. 1440–1447, Jul. 2018.

[36] 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 *Proc. 26th IEEE Int. Conf. Emerg. Technol. Factory Autom. (ETFA)*, Sep. 2021, pp. 01–08.

[37] Z. Xu and Q. Zhu, "Secure and resilient control design for cloud enabled networked control systems," in *Proc. 1st ACM Workshop Cyber-Phys. Syst.-Secur. Privacy*, Oct. 2015, pp. 31–42.

[38] F. Akbarian, W. Tärneberg, E. Fitzgerald, and M. Kihl, "A cloud-control system equipped with intrusion detection and mitigation," *Electron. Commun. EASST*, vol. 80, pp. 1–5, Sep. 2021.

[39] *Kubernetes*. Accessed: Sep. 2021. [Online]. Available: https://kubernetes.io

[40] *Ingress-NGINX*. Accessed: Sep. 2021. [Online]. Available: https://github.com/kubernetes/ingress-nginx

[41] *Prometheus-Operator*. Accessed: Sep. 2021. [Online]. Available: https://github.com/coreos/prometheus-operator

[42] M. S. Mahmoud and Y. Xia, *Cloud Control Systems: Analysis, Design and Estimation*. New York, NY, USA: Academic, 2020.

[43] F. Akbarian, W. Tärneberg, E. Fitzgerald, and M. Kihl, "Detection and mitigation of deception attacks on cloud-based industrial control systems," in *Proc. 25th Conf. Innov. Clouds, Internet Netw. (ICIN)*, Mar. 2022, pp. 106–110.

[44] F. Akbarian, E. Fitzgerald, and M. Kihl, "Intrusion detection in digital twins for industrial control systems," in *Proc. Int. Conf. Softw., Telecommun. Comput. Netw. (SoftCOM)*, Sep. 2020, pp. 1–6.
[45] D. Simon, *Optimal State Estimation: Kalman, H Infinity, and Nonlinear Approaches*. Hoboken, NJ, USA: Wiley, 2006.
[46] M. Farsi, A. Daneshkhah, A. Hosseinian-Far, and H. Jahankhani, *Digital Twin Technologies and Smart Cities*. Berlin, Germany: Springer, 2020.
[47] A. Moser, C. Appl, S. Brüning, and V. C. Hass, "Mechanistic mathematical models as a basis for digital twins," in *Digital Twins*. Cham, Switzerland: Springer, 2020, pp. 133–180.
[48] *Chaos Mesh*. Accessed: Sep. 2021. [Online]. Available: https://chaos-mesh.org/

FATEMEH AKBARIAN (Member, IEEE) received the M.S. degree in electrical engineering from Tarbiat Modares University (TMU), Tehran, Iran, in 2017. She is currently pursuing the Ph.D. degree with the Department of Electrical and Information Technology, Lund University, Lund, Sweden. Her research interests include cloud control systems, secure control, and fault-tolerant control systems.

WILLIAM TÄRNEBERG (Member, IEEE) was born in Malmö, Sweden, in 1984. He received the M.Sc. degree in electrical engineering and the Ph.D. degree from Lund University, Sweden, in 2010 and 2019, respectively.

From 2019 to 2021, he was a Postdoctoral Fellow and then a Researcher with the Department of Electrical and Information Technology, Lund University. Since 2021, he has been an Assistant Professor with the Department of Electrical and Information Technology. His research interests include self-adaptive systems, quality elastic computing, control over the cloud, industrial cloud, and 6G.

EMMA FITZGERALD (Member, IEEE) was born in Sydney, Australia, in 1986. She received the B.Sc., B.E., and Ph.D. degrees from The University of Sydney, Australia, in 2008 and 2013, respectively.

From 2014 to 2019, she was a Postdoctoral Fellow and then a Researcher with the Department of Electrical and Information Technology, Lund University, Sweden. Since 2019, she has been an Associate Professor with the Department of Electrical and Information Technology. From 2018 to 2021, she was also an Adjunct Researcher with the Warsaw University of Technology, Poland. Her research interests include network performance analysis, the Internet of Things, and beyond 5G wireless networks. In 2021, she was awarded the title of Docent (Reader) by Lund University.

MARIA KIHL (Member, IEEE) was born in Stockholm, Sweden, in 1969. She received the M.S. and Ph.D. degrees from Lund University, Sweden, in 1993 and 1999, respectively.

From 1999 to 2014, she was a Postdoctoral Fellow and then a Senior Lecturer with the Department of Electrical and Information Technology, Lund University. Since 2014, she has been a Full Professor in internetworked systems with the Department of Electrical and Information Technology. She is currently the Head of the Secure and Networked Systems Division and the Research Leader of the Networked Systems Laboratory. She has authored more than 120 peer-reviewed articles and one book. Her research interests include mission-critical networked applications, industry 4.0, and cloud control.