Detecting and Mitigating Actuator Attacks on Cloud Control Systems through Digital Twins

Akbarian, Fatemeh; Tärneberg, William; Fitzgerald, Emma; Kihl, Maria

Published in:
The 31st International Conference on Software, Telecommunications and Computer Networks (SoftCOM 2023)

DOI:
10.23919/SoftCOM58365.2023.10271648

2023

Document Version:
Publisher's PDF, also known as Version of record

Link to publication

Citation for published version (APA):
Akbarian, F., Tärneberg, W., Fitzgerald, E., & Kihl, M. (2023). Detecting and Mitigating Actuator Attacks on Cloud Control Systems through Digital Twins. In *The 31st International Conference on Software, Telecommunications and Computer Networks (SoftCOM 2023)* IEEE - Institute of Electrical and Electronics Engineers Inc.. https://doi.org/10.23919/SoftCOM58365.2023.10271648

Total number of authors:
4

Detecting and Mitigating Actuator Attacks on Cloud Control Systems through Digital Twins

Fatemeh Akbarian[1], William Tärneberg[1], Emma Fitzgerald[1] and Maria Kihl[1]
[1]*Department of Electrical and Information Technology, Lund University, Sweden*

***Abstract*—Recently, the industry has been driven to move industrial control systems to the cloud due to the significant advantages it offers in terms of storage and computing resources. However, this shift also brings forth significant security challenges. By moving control systems to the cloud, the potential for attackers to infiltrate the system and launch damaging attacks increases. These attacks can result in severe disruptions and potentially catastrophic consequences. Hence, attack detection and mitigation mechanisms are crucial for cloud control systems. In this paper, we present an approach that leverages the digital twins concept and virtual actuator method to detect and mitigate deception attacks on control signals within cloud control systems. By conducting experiments on a real testbed and subjecting it to a set of attacks, we validate the effectiveness of our solution. Our proposed method successfully detects attacks in a timely manner and keeps the plant stable, with a good performance during the attack.**

***Index Terms*—cloud control system, attack detection, attack mitigation, digital twins**

I. Introduction

In recent years, cloud computing has emerged as a transformative technology, revolutionizing the way organizations handle data and deploy their computing resources. By leveraging the power of remote servers and virtualization technologies, cloud computing offers unparalleled scalability, flexibility, and cost-efficiency. These advancements have paved the way for the control of industrial control systems over the cloud, enabling the utilization of seemingly endless computing and storage resources. With cloud-based architectures, control strategies can now be executed using more advanced algorithms and techniques, allowing controllers to tackle complex problems that are computationally demanding. Cloud control systems (CCS) leverage the cloud infrastructure, moving the control functionality to a centralized location, enabling remote monitoring, centralized control, and seamless integration with other cloud-based services, thereby enhancing operational efficiency and performance [1].

However, with the increasing adoption of cloud control systems, a new set of challenges emerges, particularly in terms of security. Connecting these systems to the cloud introduces access points for potential exploitation by malicious actors. Integrating cloud control systems with the broader network infrastructure creates opportunities for attackers, posing significant security risks. These risks include unauthorized access, data breaches, and service disruptions. Hence, it is crucial to prioritize robust security measures to protect critical infrastructure against sophisticated attack techniques [2].

Cybersecurity measures can be categorized as prevention, detection, and mitigation. Prevention reduces vulnerabilities through encryption, firewalls, and security protocols. Detection monitors for anomalies caused by attacks, while mitigation aims to minimize their impact. Detection and mitigation actions are crucial, as attackers may bypass prevention measures. In addition, implementing prevention measures like encryption for older systems like power grids can be costly due to equipment updates [3].

Different types of attacks can occur on CCSs, but this paper specifically addresses deception attacks within CCSs, focusing on their detection and mitigation. A deception attack targets the integrity of the signals sent between the cloud and the plant through the network. These signals can be either control or measurement signals, and their compromise can severely impact system performance and stability [4].

Some research has been done regarding deception attacks, and they have proposed different solutions using machine learning techniques [5], designing secure control [6], [7], using watermarking signals [8], [9], and using fault tolerant control methods [10], [11]. But, all these works have considered attacks on measurement signals.

In [12], we proposed a method to detect and mitigate attacks on control signals. We used the analytical redundancy relations (ARR) method to estimate the attack signal and reconstruct the actual control signal, effectively maintaining system stability and reliable performance during an attack. However, this method relies on the system model, and its applicability depends on the model's characteristics. Therefore, this paper introduces an alternative mitigation method that can be applied to various systems, regardless of their specific models. We aim to detect attacks on the control signal by combining the digital twins concept and the generalized likelihood ratio (GLR) method. Additionally, we employ the virtual actuator method to reconfigure the controller and mitigate the attack's impact on the system.

In the rest of this paper, first, we explain our targeted system. Next, we delve into the investigation of our proposed solution, outlining the methodology and techniques employed. Then, we illustrate the details of the experiments and evalu-

This paper was supported by the Celtic-Next project IMMINENCE funded by Vinnova, and the SSF project SEC4FACTORY under grant no. SSF RIT17-0032. The authors are part of the Excellence Center at Linköping-Lund on Information Technology (ELLIIT) strategic research area, and the Nordic University Hub on Industrial IoT (HI2OT) funded by NordForsk. Maria Kihl is partially funded by the Wallenberg AI, Autonomous Systems and Software Program (WASP) funded by the Knut and Alice Wallenberg Foundation.

ation of the proposed solution on a real testbed. Finally, we analyze and discuss the results obtained from the experiments, drawing conclusions and insights from our findings.

II. Targeted System and Testbed

A Cloud Control System integrates cloud computation with physical processes, enabling the use of advanced controllers that require substantial computation power. However, cybersecurity is a major concern for these systems. The system consists of a cloud-based controller, a physical domain with a plant, sensor(s), and actuator(s). Measurement signals (y) are transmitted from the physical domain to the cloud controller, while control signals (u) are sent back to the actuators. The network between the cloud and the physical domain serves as a potential access point for attackers to manipulate the signals, as shown in Fig. 1.

As was discussed in Section I, we consider deception attacks on control signals in which the attacker manipulates these signals. Hence, this attack can be modeled as follows:

$$\tilde{u}(k) = u(k) + a(k) \tag{1}$$

where $u_k = [u_1 \quad u_2 \quad \dots \quad u_{n_u}]$ is the control signal vector, $a_k = [a_1 \quad a_2 \quad \dots \quad a_{n_u}]$ is the attack vector, and k shows time instance. The attack vector has nonzero entries for the control signals under attack and zero values for all other control signals. In this paper, we also consider simultaneous attacks, which means that it is possible to have the attack on several control signals at the same time. Hence, for n_u control signals, there could be 2^{n_u} different attack modes. However, totally there will be $2^{n_u} - 2$ modes of attack because we assume the attacker is not able to have access to all control signals and apply attacks on them at the same time, and also, one mode is related to 2^0 in which there is no attack on control signals. So, we subtract 2 from 2^{n_u}.

III. Proposed Solution

This section describes our proposed solution to detect and mitigate deception attacks on control signals in CCSs.

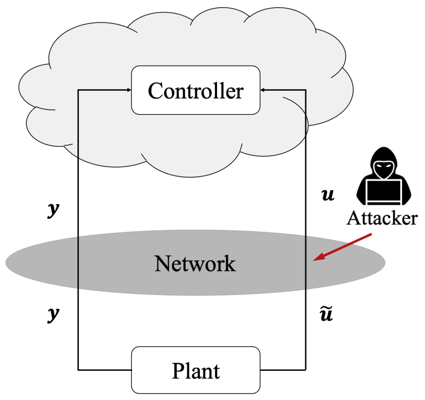

Fig. 1. CCS under actuator attack.

A. Attack Detection

1) Digital twin: For the attack detection part, we propose to use the digital twin concept. Digital twins is a rather new concept that precisely mirrors the internal behavior of the system. Hence, this fact can be used for creating a reference that by comparing the behavior of the real system with its digital twin's, any abnormal changes can be noticed.

Digital twin usually is located in cloud resources and includes simulation of all components and units of the physical part. Based on [13], as digital twins for Industrial Control System (ICS) we have a virtual replica in the cloud for each component in the physical domain. However, in cloud control systems, the controller has already been implemented in the cloud to control the real plant. Hence, to create a digital twin, as Fig.2 shows, we can have only the virtual replica of the plant and use the cloud controller for both the real plant and its digital version. In this paper, to create the virtual plant, we use the mathematical model of the plant, which can be obtained either based on the physical laws or using system identification methods.

In Fig. 2, measurement signals are sent from the real plant to the cloud controller, and control signals are transmitted to both the real and virtual plants. Both the real and the virtual plants get the same control signals, and the output of the virtual one shows what is expected from the real plant. Hence, by comparing the virtual plant's outputs $\hat{y}$ with the real plant outputs y, abnormal changes can be detected. So, residual signals r are generated as follows:

$$r(k) = y(k) - \hat{y}(k) \tag{2}$$

where subscript k denotes the time instance. In normal conditions, the outputs of the real and virtual plants should be the same, resulting in zero residual signals. However, noise in real systems can cause non-zero residuals, making it challenging to distinguish between attacks and noise. To address this, a decision system is needed to evaluate residual signals and trigger an alarm upon detecting an attack.

2) Decision system: As a decision system, we should use change detection methods to diagnose the attack from noise. The most well-known change detection methods are cumulative sum (CUSUM) and generalized likelihood ratio (GLR) algorithms. CUSUM is a test for a known magnitude of change. However, here depending on the attack signal, we can have different magnitudes of change. Hence, we suggest using GLR method in our decision system.

GLR method, which is based on the Neyman–Pearson's approach, and its aim is to distinguish between two hypotheses: $\mathcal{H}_0$—the nominal case, and $\mathcal{H}_1$—a change has taken place in the residual signals r. In normal conditions, residual signals are close to zero and have normal distribution $r = \mathcal{N}\left(\mu_0 = 0, \sigma^2\right)$ since the measurement noise on y have a normal distribution. GLR method is able to estimate not only the change time but also the change magnitude. It estimates the change magnitude $\hat{\mu}_1$ as follows:

$$\hat{\mu}_1 = \frac{1}{M} \sum_{i=k-M+1}^{k} r(i) \tag{3}$$

where M is the window size, which means that in each time interval, the last M time instants are used to calculate $\hat{\mu}_1$ at that time. Based on the estimated change magnitude, GLR decision function is calculated as follows:

$$g(k) = \frac{1}{2\sigma^2} \frac{1}{M} \left(\sum_{i=k-M+1}^{k} (r(i) - \mu_0) \right)^2 \tag{4}$$

and using the threshold h, the change can be detected using the following expression:

$$\begin{cases} \mathcal{H}_0 : g(k) \leq h \\ \mathcal{H}_1 : g(k) > h. \end{cases} \tag{5}$$

So, based on (5), false alarms occur when the system is in $\mathcal{H}_0$ but $g(k) > h$. Thus, for preventing such false alarms, the proper h can be obtained for a given probability of false alarms, P_F, by solving the following equation:

$$P_F = \int_{2h}^{\infty} \frac{1}{2^{\frac{1}{2}} \mid \Gamma\left(\frac{1}{2}\right)} X^{-\frac{1}{2}} e^{-\frac{X}{2}} dX \tag{6}$$

where $\Gamma(u)$ denotes the gamma function. There is a trade-off between the probability of false alarms and the time to detect the change. If the probability of false alarms is too small, consequently, there will be a higher threshold, and this will result in longer change detection times. A missed detection occurs when the system is in $\mathcal{H}_1$, but $g(k) \leq h$. Hence, the probability of detection, P_D, is defined based on the probability of missed detection P_M as $P_D = 1 - P_M$. After selecting h using (6), the window size M for a given probability of detection can be selected by solving the following equation:

$$P_D = \int_{2h}^{\infty} p_{\chi^2} \left(X; 1, \frac{M(\mu_1 - \mu_0)^2}{\sigma^2} \right) dX \tag{7}$$

where:

$$p_{\chi^2}(X; 1, \lambda) = \frac{1}{2} \left(\frac{X}{\lambda} \right)^{-\frac{1}{4}} e^{-\frac{X+\lambda}{2}} \mathcal{I}_{-\frac{1}{2}}(\sqrt{\lambda X}) \tag{8}$$

and $\mathcal{I}$ refers to the Bessel function of first kind.

B. Attack Mitigation

We consider a reconfiguration block to hide the attack in the mitigation part. In fact, instead of reconfiguring the controller for the attack condition, we consider a reconfiguration block to adapt the attacked system to the current controller. So, for a given control signal u_c, the attacked system should produce the same output as the normal condition y_c and send it to the controller. The controller based on y_c generates u_c.

By considering the attack on control signals, the plant under attack can be modeled as follows:

$$\begin{aligned} \dot{x}_a &= Ax_a + B_f u_a + Ed \\ y_a &= Cx_a \end{aligned} \tag{9}$$

where u_a is the control signal after applying the attack, y_a is the measurement signal generated based on u_a, and x_a is the

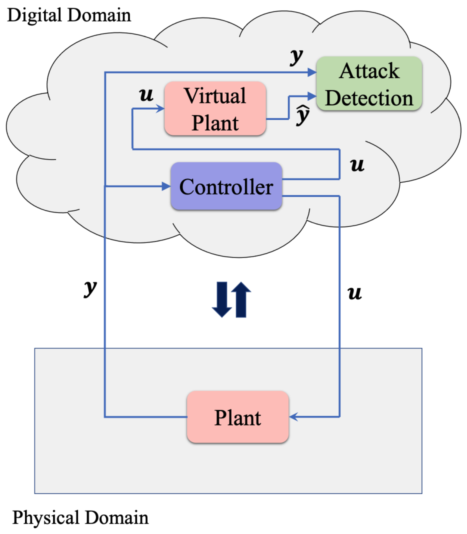

Fig. 2. Digital twins for CCS.

states of the under_attack plant. Also, d is the disturbance, A, B, E, and C are coefficient matrices in the normal system's model. B_f is matrix B in that the columns related to control signals under attack are considered zero. So this means that healthy control signals should control the system. If (A, B_f) is controllable, the reconfigured control input accesses the entire state vector and the full dynamics of the nominal system can be shaped. The controllability of (A, B_f) can be checked as follows. (A, B_f) is controllable if:

$$\text{rank} \begin{bmatrix} B_f & AB_f & A^2B_f & \dots & A^{n-1}B_f \end{bmatrix} = p \tag{10}$$

where p is the dimension of the matrix $A \in R^{p \times p}$. But, if (A, B_f) is not controllable, at least it should be stabilizable in which condition approximation of nominal output can be shaped [14]. Then we can design the virtual actuator as follows:

$$\begin{aligned} \dot{x}_v &= A_v x_v + B_v u_c \\ u_a &= M x_v + N u_c \\ y_c &= C_v x_v + y_a \end{aligned} \qquad \text{where:} \qquad \begin{aligned} A_v &= A - B_f M \\ B_v &= B - B_f N \\ C_v &= C \\ N &= B_f^+ B \end{aligned}$$

and M is chosen such that $A - B_f M$ is Hurwitz. Using this virtual actuator, we only use healthy control signals and change them such that only by using them we can control the system well.

IV. Experiment

In this section, we evaluate our approach using a ball and beam process as the plant. We deploy an MPC controller within the cloud to control it. By deploying the digital twin and our detection and mitigation approach in the cloud, we demonstrate the resilience of our cloud control system against deception attacks on control signals. The following explains the ball and beam system and our evaluation methods in detail.

A. 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 declarative configuration and automation [15]. The cluster has been equipped with an nginx ingress [16] and Prometheus operator [17]. The nginx ingress is exposed using the K8S NodePort paradigm. We use this K8S cluster to implement our controller and attack detection and mitigation algorithm.

B. Process

In this paper, we examine our proposed method using a ball and beam system. This system consists of a long beam and an electric motor that tilts the beam, causing a ball to roll back and forth on top of it. Without a controller, the system is open-loop unstable, resulting in the ball falling off the beam. The controller's goal is to maintain the ball at a set-point on top of the beam by tilting it using the electric motor. In this system, the beam length is $1.1m$, and the ball's allowed position range is $[-0.55m, 0.55m]$. In all experiments, the setpoint for the ball's position is $-0.3m$.

The attacker aims to drive the ball out of the allowed range, causing it to fall off the beam. Additionally, if the attacker moves the ball from its set-point but keeps it on the beam, it leads to increased costs and decreased efficiency. Therefore, our objective in this paper is to maintain the ball on the beam at the exact set-point, even in the presence of an attack.

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. Also, any attacks can make it unstable easily. 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.

This system has three measurement signals: the ball's position, the ball's speed, and the beam's angle. One control signal sets the beam's speed to adjust the ball's position. This system can be modeled as follows:

$$\dot{\boldsymbol{x}}(t) = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & -\frac{5g}{7} \\ 0 & 0 & 0 \end{bmatrix} \boldsymbol{x}(t) + \begin{bmatrix} 0 \\ 0 \\ 0.44 \end{bmatrix} \boldsymbol{u}(t)$$
$$\boldsymbol{y}(t) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \boldsymbol{x}(t) \tag{11}$$

To design the controller for this system, we discretize it with a sampling time of $T_s = 0.05s$ and employ the Linear–quadratic regulator (LQR). The LQR generates the control signal as:

$$u(k) = -K(y(k) - s(k)) \tag{12}$$

Here, $K = [-31.0027, -13.9077, 20.4937]$ represents the gain vector, y_k denotes the measurement signal, and s_k denotes the setpoint. The controller is deployed as a pod in a Kubernetes cluster serving as the cloud.

Our mitigation method, discussed in Section III-B, is designed for multi-input systems. However, the ball and beam system is a single-input system, deliberately chosen to demonstrate the applicability of our mitigation method even in such cases. We can transform this system into a multi-input system by modifying it. Firstly, we multiply the gain vector by K_a in the controller, expanding the dimension of u as follows:

$$K_{new} = K_a K = \begin{bmatrix} 1/2 \\ 1/3 \\ 1/5 \\ 1/4 \end{bmatrix} K \tag{13}$$

Hence, the controller will generate four control signals instead of one:

$$u_{new}(k) = -K_{new}(y(k) - s(k)) \tag{14}$$

Based on K_{new}, the ball and beam system's model in (11) should be adjusted to four control signals. For this, B matrix should be replaced with B_{new} such that meets following condition:

$$Bu = B_{new} u_{new} = B_{new} K_a u \longrightarrow B = B_{new} K_a \tag{15}$$

Using B_{new}, (11) is changed to a multi-input system as follows:

$$\dot{\boldsymbol{x}}(t) = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & -\frac{5g}{7} \\ 0 & 0 & 0 \end{bmatrix} \boldsymbol{x}(t) + \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0.5 & 0.66 & 1.1 & -1 \end{bmatrix} \boldsymbol{u}(t)$$
$$\boldsymbol{y}(t) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \boldsymbol{x}(t) \tag{16}$$

Now, for this multi-input system, we can employ a virtual actuator to mitigate attacks. However, the actuator in the real system can only apply a single control signal to adjust the beam's speed. Therefore, we need to recalculate the main control signal using the four control signals received on the plant side. By utilizing equation (15) and knowing the values of B, B_{new}, and u_{new}, we can calculate u as $u = B^{-1} B_{new} u_{new}$.

C. Digital Twin

As explained in Section III-A1, the digital twin is deployed inside the cloud and continuously follows its physical counterpart. We use the nonlinear model of the ball and beam system that is more precise to create its digital twin as follows:

$$\begin{aligned} x_3(k+1) &= 0.44 T_s u(k) + x_3(k) \\ P &= \frac{5}{7} g \sin(x_3(k+1)) \\ N &= \frac{5}{7} 0.44^2 u^2(k) \\ Q &= T_s x_2(k) + x_1(k) \\ x_2(k+1) &= \frac{NQ - P}{1 - BT_s^2} T_s + x_2(k) \\ x_1(k+1) &= T_s x_2(k+1) + x_1(k) \\ x(k+1) &= [x_1(k+1), x_2(k+1), x_3(k+1)]^T \end{aligned} \tag{17}$$

$g = 9.80665$ is the gravitational constant, $u(k)$ is the control signal, and $x(k)$ is the state vector consisting of $x_1(k)$ (ball's position), $x_2(k)$ (ball's speed), and $x_3(k)$ (beam's angle). The digital twin is implemented in Python and deployed as a pod in a K8S cluster. The control signal generated by the LQR

TABLE I
DIFFERENT MODES OF ATTACK.

	Mode 1	Mode 2	Mode 3	Mode 4	Mode 5	Mode 6	Mode 7	Mode 8	Mode 9	Mode 10	Mode 11	Mode 12	Mode 13	Mode 14
Attack on u_1	×	-	-	-	×	×	×	-	-	-	×	×	×	-
Attack on u_2	-	×	-	-	×	-	-	×	×	-	×	×	-	×
Attack on u_3	-	-	×	-	-	×	-	×	-	×	×	-	×	×
Attack on u_4	-	-	-	×	-	-	×	-	×	×	-	×	×	×

controller is utilized for both the cloud-based digital twin and the physical system.

D. Applying Attacks on the System

In this paper, we assume attacks occur on control signals and change their value. We also assume simultaneous attacks can be applied to control signals. Based on Section IV-B, the plant in our testbed has four control signals, and based on that the attack occurs on which of them, we can define $2^4 - 2 = 14$ different modes as shown in Table I. We subtract 2 from 2^4 because we disregard the case in which there is no attack on the control signals and also the case in which we have an attack on all control signals since we assume the attacker is not able to have access to all control signals at the same time.

E. Evaluation of Attack Detection Method

In the initial phase of the experiment, we evaluate the attack detection part. In (1), based on the mode of attack, if the elements of the attack vector possess higher values, it induces rapid changes in the position of the ball, thereby impacting the system. However, such attacks are easily detectable. Hence, to properly assess the detection method capabilities, we consider an attack in the form of noise added to the actual control signal. Detecting this type of attack poses significant challenges, as it can resemble normal system noise, rendering it difficult to distinguish from genuine system behavior. So, based on the mode of the attack that we consider, we consider the non-zero elements of the attack vector in (1) as noise with normal distribution $a_i = \mathcal{N}(0.1, 0.0016)$ that is applied to the control signals at the 700th sample.

F. Evaluation of Attack Mitigation Method

In the second part of the experiments, to evaluate our mitigation part, we apply a powerful attack on the system, such that the attack can make the system unstable immediately. So, based on the mode of the attack that we consider, we consider the non-zero elements of the attack vector in (1) as normal distribution $a_i = \mathcal{N}(9, 0.0016)$ that is applied to the control signals at the 700th sample. Despite the easy detectability of this attack, we use that here in our experiment to assess the efficacy of our proposed mitigation method.

Also, as a performance metric, we use Integral Absolute Error(IAE) as follows to compare the control performance in normal conditions, in attack conditions when we have mitigation, and in attack conditions when we do not have any mitigation.

$$IAE = \sum_{k=0}^{T} |y(k) - s(k)|, \tag{18}$$

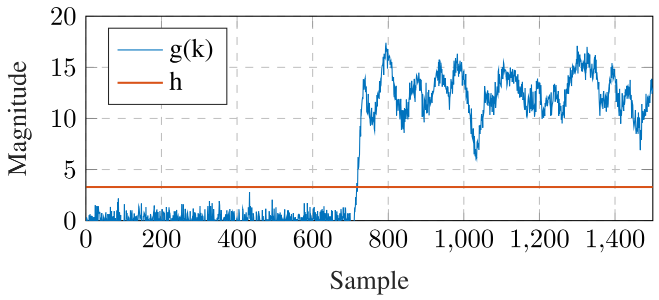

Fig. 3. Detecting attack in mode 3 using the proposed method.

where $y(k)$ is the position signal, and $s(k)$ is the set-point for the ball's position on the beam. We have chosen IAE as a performance metric for evaluating our mitigation method because the control objective is tracking the reference in our control system (ball and beam system).

V. RESULTS AND DISCUSSION

This section presents the results of the evaluation of attack detection and mitigation algorithms.

First, Based on Section IV-E, we apply noise-based attacks to control signals in different modes (Table I). Fig. 3 demonstrates the operation of our attack detection method for mode 3, targeting u_3. The red line represents the threshold $h = 3.3174$ (determined from $P_F = 10^{-2}$ using Eq. (6)), while the blue line depicts the decision function $g(k)$ surpassing the threshold at the 717th sample. With a sampling time of $T_s = 50$ ms, the attack is detected in approximately 0.85 s. We repeat this experiment for all 14 modes, measuring the detection time. Fig. 4 displays the results, demonstrating that the attack is detected in less than 2 s in all cases, confirming the effectiveness of our method.

In the second experiment phase (Section IV-F), we assess our mitigation approach by applying a strong attack that can immediately destabilize the system. Fig. 5 illustrates the results of applying the attack in mode 1. Without the mitigation

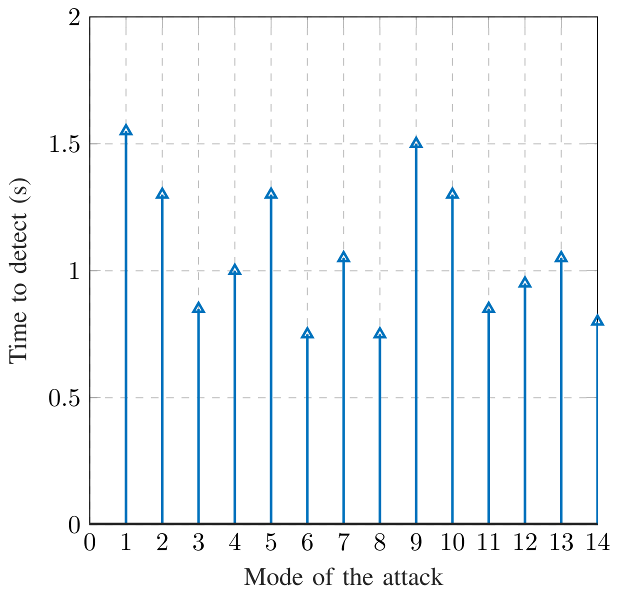

Fig. 4. The time it takes to detect different modes of attacks.

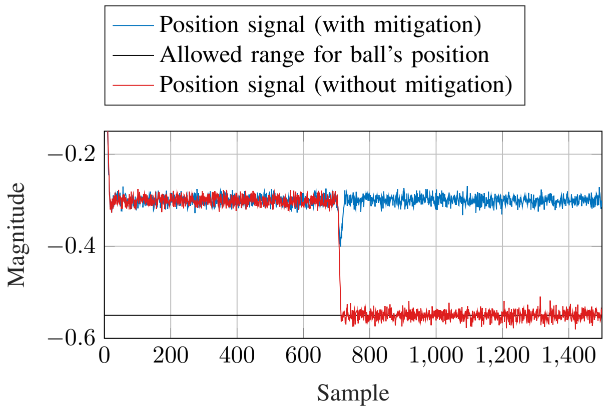

Fig. 5. Mitigating attack on Mode 1.

method, the system quickly becomes unstable, as shown by the red curve. The position signal at the 713th sample exceeds the permissible range (black line), causing the ball to fall off the beam. However, a different outcome is observed when our mitigation method is implemented (blue curve). The attack is detected shortly after initiating at the 700th sample, triggering the mitigation mechanism. The ball is then moved back to the setpoint, effectively saving the system. We will now extend our analysis to other attack modes and calculate the IAE (based on Eq. (18)) to evaluate the control performance in the presence of attacks. Fig. 6 presents the IAE for all 14 attack modes, considering the powerful attack described in Section IV-F.

In Fig. 6, the bars represent IAE for different conditions, measured until the 750th sample. The red bars denote the normal condition without attacks. The blue bars show the IAE for the attack condition without mitigation, which exhibits higher values compared to the normal condition. In contrast, the purple bars display the IAE for the attack condition with our mitigation approach. Across all attack modes, the mitigated condition using our proposed security framework consistently yields significantly lower IAE compared to the case without mitigation. Furthermore, the IAE in the mitigated condition closely resembles the IAE in the normal condition.

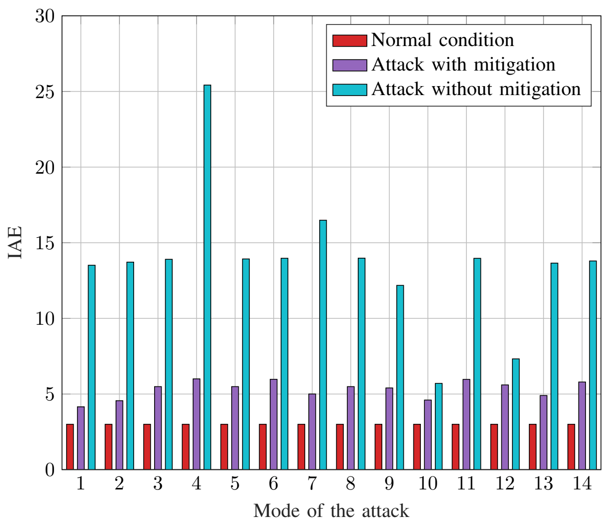

Fig. 6. IAE for each mode of the system.

This highlights the effectiveness of our mitigation strategy in maintaining plant stability and achieving good control performance in the presence of attacks. It demonstrates our solution's capability to counteract the adverse effects of attacks, ensuring system stability and control quality.

VI. Conclusion

This paper introduces an innovative method for detecting and mitigating attacks on control signals in cloud control systems. Our approach utilizes digital twins to swiftly detect deception attacks on control signals. We have also developed virtual actuators to effectively mitigate detected attacks, ensuring system stability and maintaining performance even during an attack. To validate our solution, we implemented it on a real testbed and conducted thorough evaluations. The results unequivocally demonstrate the effectiveness of our approach in detecting and mitigating attacks on control signals in cloud control systems.

References

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

[2] Y. Xia, Y. Zhang, L. Dai, Y. Zhan, and Z. Guo, "A brief survey on recent advances in cloud control systems," *IEEE Transactions on Circuits and Systems II: Express Briefs*, 2022.

[3] R. Mattioli and K. Moulinos, "Communication network interdependencies in smart grids," *EUA FNAI Security, Ed., ed. EU: ENISA*, 2015.

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

[5] 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 Journal of Automatica Sinica*, vol. 9, no. 3, pp. 377–391, 2021.

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

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

[8] A. Naha, A. Teixeira, A. Ahlén, and S. Dey, "Deception attack detection using reduced watermarking," in *2021 European Control Conference (ECC)*. IEEE, 2021, pp. 74–80.

[9] I. Bessa, C. Trapiello, V. Puig, and R. M. Palhares, "Dual-rate control framework with safe watermarking against deception attacks," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, 2022.

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

[11] F. Akbarian, W. Tärneberg, E. Fitzgerald, and M. Kihl, "Attack resilient cloud-based control systems for industry 4.0," *IEEE Access*, vol. 11, pp. 27 865–27 882, 2023.

[12] ——, "Detection and mitigation of deception attacks on cloud-based industrial control systems," in *2022 25th Conference on Innovation in Clouds, Internet and Networks (ICIN)*. IEEE, 2022, pp. 106–110.

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

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

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

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

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