Journal of Internet Services and Applications

RESEARCH Open Access

A controller design for mitigation of passive system identification attacks in networked control systems

Alan O. de Sá[1,2*], Luiz F. R. da Costa Carmo[1,3] and Raphael C. S. Machado[3,4]

Abstract

The literature regarding attacks in Networked Control Systems (NCS) indicates that covert and accurate attacks must be designed based on an accurate knowledge about the model of the attacked system. In this context, the literature on NCS presents the Passive System Identification attack as a metaheuristic-based tool to provide the attacker with the required system models. However, the scientific literature does not report countermeasures to mitigate the identification process performed by such passive metaheuristic-based attack. In this sense, this work proposes the use of a randomly switching controller as a countermeasure for the Passive System Identification attack, in case of failure of other conventional security mechanisms – such as encryption, network segmentation and firewall policies. This novel countermeasure aims to hinder the identification of the controller, so that the model obtained by the attacker is imprecise or ambiguous, in such a way that the attacker hesitates to launch covert or model-dependent attacks against the NCS. The simulation results indicate that this countermeasure is capable to mitigate the mentioned attack at the same time that it performs a satisfactory plant control.

Keywords: Networked control system (NCS), Cyber-physical systems, Security, System identification attacks, Switching controller

1 Introduction

A Networked Control System (NCS) is constituted by a physical plant whose dynamics is controlled by a digital controller – i.e. a computational system – through a communication network which, indeed, integrates the cyberspace to the physical domain. The integration of controllers and physical processes via communication networks aims to provide these systems with better operational and management capabilities, as well as reduce costs. By virtue of these advantages, the number of NCSs applied to industrial processes and critical infrastructure systems is increasing [1–10]. A diagram of an NCS is depicted in Fig. 1, wherein $G(z)$ is the transfer function of the plant, $C(z)$ is the control function executed by the controller and both devices are interconnected through the forward and a feedback streams. The forward stream carries the control signals from the controller to the plant's actuators. The feedback stream, in turn, carries the sensed data from the plant to the controller.

Despite the advantages provided by the NCSs, the integration of controllers and physical plants through a communication network also exposes such control systems to threats originated in the cyber domain. In this context, there is a research effort to characterize vulnerabilities and propose security solutions for NCSs.

Recent researches on the security of NCSs demonstrate the development of a set of sophisticated attacks [6, 11, 12] that, to be covert and accurate, are designed based on the models of the attacked system. For instance, in [12, 13][1], the authors present an attack where false data is injected in the communication process of an NCS to degrade the service performed by a plant. The changes driven by this attack are dimensioned so that the modifications in the

*Correspondence: alan.oliveira.sa@gmail.com
[1] Institute of Mathematics/NCE, Federal University of Rio de Janeiro, Av. Athos da Silveira Ramos, 274, 68.530 Rio de Janeiro, Brazil
[2] Admiral Wandenkolk Instruction Center, Brazilian Navy, Enxadas Island, Guanabara Bay, Rio de Janeiro, Brazil
Full list of author information is available at the end of the article

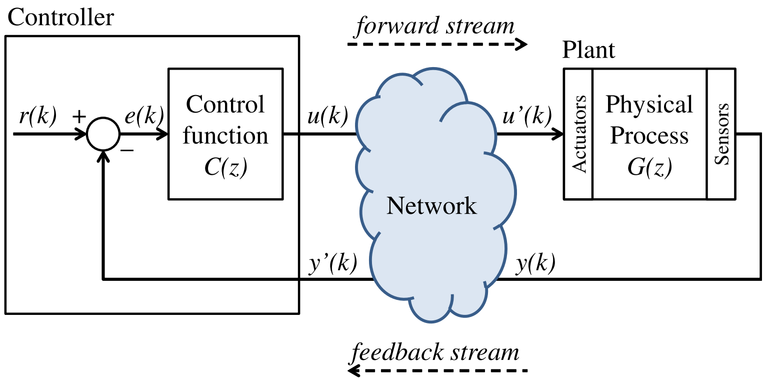

Fig. 1 Networked control system (NCS) [12]

plant's behavior are physically difficult to be perceived. For this reason, this attack is classified as physically covert [12]. To ensure that the attack proposed in [12] is physically covert, the authors indicate that the attacker must plan the offensive based on an accurate knowledge about the system dynamics – otherwise the consequences of the attack may be unpredictable. In this case, the unpredictable behavior of the plant can provide physical evidence that it is being manipulated, drawing the attention to the possibility of a cyber-physical attack.

One possible way to obtain such knowledge about the NCS is through conventional intelligence operations, performed to collect information regarding the design of the system. Another way to gather information about the targeted system is through a Cyber-Physical Intelligence attacks [12]. To this end, the authors of [12] propose a metaheuristic-based Passive System Identification attack, which aims to collect information about the plant's transfer function $G(z)$ and the controller's control function $C(z)$ of an NCS. As shown in Fig. 2 (draw based on the

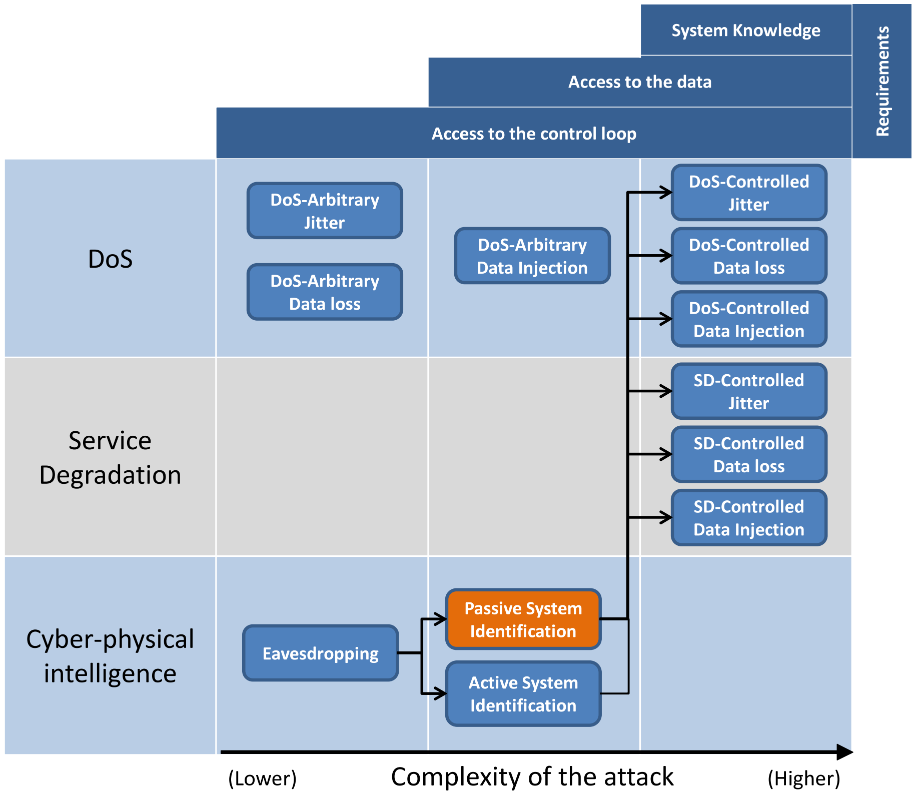

Fig. 2 Classification and requirements of cyber-physical attacks in NCSs

taxonomy proposed in [12]), the Passive System Identification attack constitutes a path to build sophisticated model-dependent attacks, once they are capable to provide the attacker with the required system knowledge. Indeed, the results of [12] demonstrate the effectiveness of the Passive System Identification attack in supporting the design of covert/model-dependent attacks.

Although the authors of [12] encourage the development of countermeasures for the Passive System Identification attack, the scientific literature – to the best of our knowledge – does not report countermeasures to mitigate the identification process performed by such passive metaheuristic-based attack. In this sense, this work aims to discuss and propose a countermeasure for the mentioned attack.

The straightforward countermeasure to prevent the success of a System Identification attack in an NCS is to avoid unauthorized access to the control loop using, for example, network segmentation, demilitarized zones (DMZ), firewall policies and implementing specific network architectures, such as recommended in [14]. A complementary countermeasure – in case the attacker is capable to access the control loop – is to hinder the access to the data flowing in the NCS using, for example, symmetric-key encryption algorithms, hash algorithms and a timestamp strategy to form a secure transmission mechanism between the controller and the plant, as proposed in [15]. However, when the mentioned countermeasures fail and the attacker gain access to the data flowing in the NCS, the alternative to prevent the attacker to obtain the model of the system is to hinder the analysis of the captured data – i.e. make the System Identification algorithm inaccurate/ineffective.

One possible strategy to cause difficulties to the System Identification algorithm is to have, in the NCS, specific control functions that are, at the same time, harder to be identified and capable to control the plant. Based on this reasoning, the contribution of this work is the proposal of a randomly switching controller design as a feasible countermeasure to mitigate the Passive System Identification attack proposed in [12]. As far as we know, there is no other countermeasure reported in the literature that mitigates the Passive System Identification attack by hindering the analysis of signals captured from the NCS.

The rest of this paper is organized as follows: First, in Section 2, some related works are presented. Later, in Section 3, the Passive System Identification attack and a subsequent Data Injection attack are described, in order to provide the underlying information necessary to comprehend the countermeasure proposed in this paper. Then, in Section 4, the switching controller is presented and discussed as a countermeasure for the Passive System Identification attack. After that, Section 5 presents simulation results, where the performance of the switching controller is analyzed from the countermeasure and control perspectives. Finally, in Section 6, some conclusions and possible future works are presented.

2 Related works

The launch of cyber-physical attacks in real world systems, such as the case of the Stuxnet [16] worm, raised the concern of governments and NCS owners, and is motivating the research on cybersecurity of industrial and critical infrastructure facilities. In this context, recent studies demonstrate the development of a set of sophisticated attacks that, to achieve a high level of covertness and accuracy, rely on the knowledge about the model of the attacked system. As recognized by the literature on NCS [12, 17], System Identification attacks are considered a key step in the development of those sophisticated attacks. So, this section presents a review on attacks in NCSs, giving special attention to the role that System Identification attacks play in the context of the cybersecurity of these control systems.

In [18], the authors evaluate the impact of delay jitter and packet loss in an NCS under a Denial of Service (DoS) attack. The conception of such DoS attack does not take into account the models of the controller and physical plant of the attacked NCS (i.e. these models are not known by the attacker). Therefore, to affect the physical process, the attacker arbitrarily floods the network, causing jitter and packet loss in the communication links of the NCS. In this tactic, the excess of packets in the network may reveal the attack, allowing the implementation of countermeasures such as packet filtering [18] or blocking the malicious traffic on its origin [19]. Additionally, as stated in [12], the arbitrary intervention in a system which the models are unknown may lead the plant to an extreme physical behavior, which is not desired if a physically covert [12] attack is intended.

In [4], the authors demonstrate an attack where false signals are transmitted to the controller and the actuator of an NCS. The false signals are randomly generated by the attacker, aiming to cause the instability of the plant (a DC motor). To evaluate this arbitrary data injection attack, the authors propose a testbed for Supervisory Control and Data Acquisition (SCADA) system, using TrueTime (a MATLAB/Simulink based tool). Such arbitrary data injection attack does not require a previous knowledge about the models of the plant and its controller. Therefore, the desired physical effect and the covertness of the attack cannot be ensured due to the unpredictable consequences of the injection of random false signals in a system which the model is not known.

In [20], the authors analyze a wide variety of attacks in NCSs and establish the requirements for the attacks in terms of model knowledge, disclosure and disruption resources. In their work, it is stated that the design of covert attacks requires a high level of knowledge about the model of the attacked system. In [6, 11, 21], examples of covert attacks that agree with the statement provided in [20] are proposed and analyzed. In [11, 21], the attacker, acting as a man-in-the-middle (MitM), injects false data in the forward stream of the NCS to take control of the plant. Then, to make the attack covert, the attacker uses the model of the attacked plant to compute the data injected in the feedback stream. The covertness of the attack proposed in [21] is analyzed from the perspective of the signals arriving at the controller and, as demonstrated in [11], it depends on the difference between the actual model of the plant and the model known by the attacker. In [6], the attacker, aware of the model of the NCS, injects data in its communication links to covertly steal water from the Gignac canal system located in Southern France.

In [6, 11, 20, 21], although the attacks are designed based on the models of the NCS, the authors do not describe how these models are obtained by the attacker. It is just stated that the models, used for the design of the covert/model-dependent attacks, are previously known by the attacker. In order to fill this gap, [12] and [17] propose two new kinds of attack to estimate the models of the attacked system: the Passive System Identification attack [12]; and the Active System Identification attack [17]. As shown in Fig. 2 – and, according to the taxonomy proposed in [12] –, these attacks belong to the category of Cyber-physical Intelligence attacks.

The Passive System Identification attack [12] – formerly referred to as System Identification attack[2] – does not need to inject signals in the NCS to estimate its models. However, the effectiveness of the Passive System Identification attack depends on the occurrence of events – not controlled by the attacker – to produce signals that carry meaningful information for the system identification algorithm. This attack passively estimates the transfer functions of both controller and plant by simply eavesdropping the forward and the feedback streams of the system. On the other hand, the Active System Identification attack constitutes an alternative to the Passive System Identification attack, in situations where the attacker cannot wait so long for the occurrence of such meaningful signals. In the Active System Identification attack, as described in [17], the attacker estimates the open-loop transfer function of the NCS by injecting an attack signal and eavesdropping its response at a single point of interception.

A synthesis of the attacks referred in this section is presented in Table 1. Based on these works, it is possible to verify how useful may be a System Identification attack for the design of covert/model-dependent attacks in NCSs. However, in the scientific literature, we still do not find specific countermeasures to mitigate the identification process performed by the attack proposed in [12]. In this context, this work proposes a countermeasure to mitigate such metaheuristic-based Passive System Identification attack, even when the attacker gets access to the data that is transmitted in the NCS.

3 Covert attack for service degradation

For the sake of completeness, this section describes the attack proposed in [12], in order to provide the information necessary to comprehend the countermeasure proposed in the present work. The attack consists of the joint operation of two attacks: the Passive System Identification Attack, detailed in Section 3.1; and the SD-Controlled Data Injection attack (model-dependent), detailed in Section 3.2. Section 3.3 presents simulation data that demonstrate the effectiveness of the Passive System Identification attack when supporting the design of SD-Controlled Data Injection attacks. These data, obtained from [12], are used as a reference for the evaluation of the proposed countermeasure.

3.1 Passive system identification attack

The Passive System Identification attack, proposed in [12], is intended to assess the coefficients of the plant's transfer function $G(z)$ and the controller's control function $C(z)$ of an NCS. To do so, the attack is modeled as an optimization problem, where the transfer function of the attacked device – be it a controller or plant – is estimated by minimizing a specific fitness function. This modeling is explained in Section 3.1.1. To minimize the mentioned fitness function, the attack uses the Backtracking Search Optimization Algorithm (BSA) [22], briefly described in Section 3.1.2.

3.1.1 Modeling the passive system identification attack as an optimization problem

If the input $i(k)$ and output $o(k)$ signals of an attacked device are known, the model of such device can be assessed by applying the known $i(k)$ in an estimated model, which must be adjusted until its estimated output $\hat{o}(k)$ converges to $o(k)$. In the present attack, the estimated model of the attacked device is iteratively adjusted by the BSA, that minimizes the fitness function herein presented, until the estimated model converges to the actual model of the real device.

To establish the fitness function, firstly, it must be considered a generic LTI system, whose transfer function $Q(z)$ is represented by (1):

$$Q(z) = \frac{O(z)}{I(z)} = \frac{a_n z^n + a_{n-1} z^{n-1} + \ldots + a_1 z^1 + a_0}{z^m + b_{m-1} z^{m-1} + \ldots + b_1 z^1 + b_0}, \quad (1)$$

Table 1 Synthesis of the related attacks

Attack	Method	Knowledge about the system?	How the knowledge is obtained?
Stuxnet *worm* [16]	Modifications in the	Yes	Experiments in a real system
Long, et al. [18]	*Jitter* and packet loss	None	N/A
Farooqui, et al. [4]	Data injection	None	N/A
Smith [11, 21]	Data injection	Yes	Not described
Teixeira [20]	Packet loss	None	N/A
	Data injection	Yes	Not described
Amin [6]	Data injection	Yes	Not described
SD-Controlled [12]	Data injection	Yes	Passive system identification attack
de Sá, et al. [17]	Data injection [a]	Yes	Active system identification attack

[a] In [17], the data injection is not used to cause the disruption or degradation of the plant. The data is injected in the NCS to support the Active System Identification attack

wherein $I(z)$ is the input of the system, $O(z)$ is the output of the system, n and m are the order of the numerator and the denominator, respectively, and $[\,a_n, a_{n-1}, \ldots a_1, a_0]$ and $[\,b_{m-1}, b_{m-2}, \ldots b_1, b_0]$ are the coefficients of the numerator and the denominator, respectively, that are intended to be found by the Passive System Identification attack. Also, it must be considered that $i(k)$ and $o(k)$ represent the sampled input and output of the system, respectively, where $I(z) = \mathcal{Z}[\,i(k)]$, $O(z) = \mathcal{Z}[\,o(k)]$, k is the number of the sample and $\mathcal{Z}$ represents the Z-transform operation.

In this Passive System Identification attack, $i(k)$ and $o(k)$ are firstly captured by an eavesdropping [23, 24] attack, during a monitoring period T. To deal with the eventual loss of samples, that may not be received by the attacker during T, the algorithm holds the value of the last received sample, according with (2), wherein $x(k)$ can either be $i(k)$ or $o(k)$:

$$x(k) = \begin{cases} x(k-1) & \text{if the sample } k \text{ is lost;} \\ x(k) & \text{otherwise.} \end{cases} \tag{2}$$

Then, after acquiring $i(k)$ and $o(k)$, the captured $i(k)$ is applied to the input of an estimated model, that is described by a transfer function whose coefficients $[\,a_{n,j}, a_{n-1,j}, \ldots a_{1,j}, a_{0,j}, b_{m-1,j}, b_{m-2,j}, \ldots b_{1,j}, b_{0,j}]$ are the coordinates of an individual j of the BSA. The application of $i(k)$ to the input of the estimated model results in an output signal $\hat{o}_j(k)$. After obtaining $\hat{o}_j(k)$, the fitness f_j of the individual j is computed comparing the output $o(k)$ – captured from the attacked device – with the output $\hat{o}_j(k)$ of the estimated model, according with (3):

$$f_j = \frac{\sum\limits_{k=0}^{N} (o(k) - \hat{o}_j(k))^2}{\mathbb{K}}, \tag{3}$$

wherein $\mathbb{K}$ is the number of samples that exist during the monitoring period T. Note that, if the attacker does not lose any sample of $i(k)$ and $o(k)$ during T, then $\min f_j = 0$ when $[\,a_{n,j}, a_{n-1,j}, \ldots a_{1,j}, a_{0,j}, b_{m-1,j}, b_{m-2,j}, \ldots b_{1,j}, b_{0,j}] = [\,a_n, a_{n-1}, \ldots a_1, a_0, b_{m-1}, b_{m-2}, \ldots\ b_1, b_0]$, i.e. when the estimated model converges to the actual model of the attacked device.

It is possible to establish an analogy between this System Identification attack and the Known Plaintext cryptanalytic attack [25], wherein $i(k)$ and $o(k)$ correspond to the plaintext and ciphertext, respectively, the form of the generic transfer function $Q(z)$ corresponds to the encryption algorithm and the actual coefficients of $Q(z)$ corresponds to the secret key.

3.1.2 Backtracking search algorithm

In this section, the basic concepts of the BSA are briefly described, in order to provide a clear comprehension regarding the parameters of the algorithm that are adjusted for the attack. The BSA is a bio-inspired metaheuristic that searches for solutions of optimization problems using the information obtained by past generations – or iterations. According to [22], its search process is metaphorically analogous to the behavior of a social group of animals that, at random intervals returns to hunting areas previously visited for food foraging. The general, evolutionary like, structure of the BSA is shown in Algorithm 1.

Algorithm 1: BSA

```
begin
    Initialization;
    repeat
        Selection-I;
        Generate new population
            Mutation;
            Crossover;
        end
        Selection-II;
    until Stopping Condition;
end
```

At the Initialization stage, the algorithm generates and evaluates the initial population $\mathcal{P}_0$ and sets the historical population $\mathcal{P}_{hist}$. The latter constitutes the BSA's memory that, in the Selection-I stage, is updated with historical coordinates visited by the individuals.

During the first selection stage (Selection-I), the algorithm randomly determines, based on a uniform distribution U, whether the current population $\mathcal{P}$ should be kept as the new historical population, and thus replace $\mathcal{P}_{hist}$ (i.e. if $a < b \mid a, b \sim U(0,1)$, then $P_{hist} = P$). Subsequently, at every iteration, it shuffles the individuals of $\mathcal{P}_{hist}$ (having $\mathcal{P}_{hist}$ been replaced or not).

The mutation operator creates $\mathcal{P}_{mod}$, which is the preliminary version of the new population $\mathcal{P}_{new}$). It does so according to (4):

$$\mathcal{P}_{mod} = \mathcal{P} + \eta \cdot \Gamma(\mathcal{P}_{hist} - \mathcal{P}), \qquad (4)$$

wherein η is empirically adjusted through simulations and $\Gamma \sim \mathbb{N}(0,1)$, with $\mathbb{N}$ being a normal standard distribution. Thus, $\mathcal{P}_{mod}$ is the result of the movement of $\mathcal{P}$'s individuals in the directions established by vector $(\mathcal{P}_{hist} - \mathcal{P})$ and η controls the displacements' amplitude.

In order to create the final version of $\mathcal{P}_{new}$, the crossover operator randomly combines, also following a uniform distribution, individuals from $\mathcal{P}_{mod}$ and others from $\mathcal{P}$.

At the second selection stage (Selection-II), the algorithm firstly evaluates the individuals of $\mathcal{P}_{new}$ using the fitness function f_j (3). After that, individuals of $\mathcal{P}$ (i.e. individuals before applying the mutation and crossover operators) are replaced by individuals of $\mathcal{P}_{new}$ (i.e. individuals obtained after mutation and crossover) with better fitness. Hence, $\mathcal{P}$ includes only new individuals that evolved. While the stopping condition has not yet been reached, the algorithm iterates. Otherwise, it returns the best solution found.

Note that the algorithm has two parameters that are empirically adjusted: the size $|\mathcal{P}|$ of its population $\mathcal{P}$; and η, that establishes the amplitude of the movements of the individuals of $\mathcal{P}$. The parameter η must be adjusted to assign to the algorithm good exploration and exploitation capabilities. With these parameters adjusted, the BSA is used to search for the global minimum of the fitness function described in Section 3.1.1 and, therefore, discover the model of the attacked device.

3.2 SD-Controlled data injection attack

The SD-Controlled Data Injection attack is a model-dependent attack, which the purpose is to reduce the MTBF of the plant and/or reduce the efficiency of the physical process that it performs, by inserting false data in the control loop of the NCS. At the same time, this attack is designed to be physically covert [12].

One way to degrade a physical service is through the induction of an overshoot during the transient response of a plant. The overshoots, or peaks occurred when the system exceeds the targeted value during the transient response, can cause stress and possibly damage physical systems such as mechanical, chemical and electromechanical systems [26, 27]. Additionally, once they occur in a short period, the overshoots are difficult to be noticed by a human observer. Another way to degrade the service of a plant is causing a constant steady state error on it, i.e. producing a constant error when $t \to \infty$. A low proportion steady state error, besides being difficult to be perceived by a human observer, may reduce the efficiency of the physical process or, occasionally, stress and damage the system in the mid/long term.

In the SD-Controlled Data Injection attack, to achieve either of the two mentioned effects, i.e. an overshoot or a constant steady state error, the attacker interfere in the NCS's communication process by injecting false data into the system in a controlled way. To do so, the attacker act as a MitM that executes an attack function $M(z)$, as presented in Fig. 3, wherein $U'(z) = M(z)U(z)$, $U(z) = \mathcal{Z}[u(k)]$ and $U'(z) = \mathcal{Z}[u'(k)]$. The function $M(z)$ is designed based on the models of the plant and the controller, both obtained through the Passive System Identification attack, described in Section 3.1. The effectiveness of the attack, therefore, depends on the design of $M(z)$, which in turn depends on the accuracy of the System Identification attack. It is worth mentioning that, in Fig. 3, although the MitM is placed in the forward stream, it is also possible to perform an attack by interfering in the feedback stream of the NCS.

3.3 Performance of the covert attack for service degradation

This section presents the results of the joint operation of the Passive System Identification attack and the SD-Controlled Data Injection attack. These results, obtained from [12], demonstrate the effectiveness of the Passive System Identification attack when accomplishing its task in an NCS without the countermeasure proposed in this paper.

The attacked NCS has the same architecture of the NCS shown in Fig. 1. It consists of a Proportional-Integral (PI) controller that controls the rotational speed of a DC motor. The PI control function $C_1(z)$ and the DC motor transfer function $G(z)$ are represented by (5) and (6), respectively:

$$C_1(z) = \frac{c_{1,1}z - c_{2,1}}{z - 1} \qquad (5)$$

$$G(z) = \frac{g_1 z + g_2}{z^2 - g_3 z + g_4} \qquad (6)$$

wherein $c_{1,1} = 0,1701$, $c_{2,1} = -0,1673$, $g_1 = 0,3379$, $g_2 = 0,2793$, $g_3 = -1,5462$ and $g_4 = 0,5646$. The sample

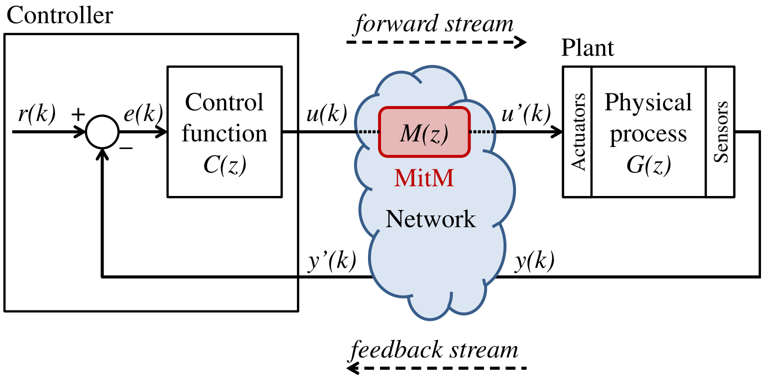

Fig. 3 MitM attack [12]

rate of the system is 50 samples/s and the set point $r(k)$ is a unitary step function.

It is considered that the structure of the Eqs. (5) and (6) are previously known by the attacker given that, as a premise, he/she knows that the target is an NCS that controls a DC motor using a PI controller. Therefore, the goal of the Passive System Identification attack is to discover $g_1, g_2, g_3, g_4, c_{1,1}$ and $c_{2,1}$.

Each time that the DC motor is turned on, the forward and the feedback streams are captured by the attacker during a period $T = 2s$. All initial conditions are considered 0, by the time that the motor is turned on. To assess $[g_1, g_2, g_3, g_4]$, the attacker considers the forward stream as the input and the feedback stream as the output of the estimated plant. In the opposite way, to assess $[c_{1,1}, c_{2,1}]$, the attacker considers the feedback stream as the input and the forward stream as the output of the estimated controller.

According to [12], in these simulations, the BSA population has 100 individuals and $\eta = 1$. To assess the coefficients of the controller $[c_{1,1}, c_{2,1}]$, the algorithm was executed for 600 iterations. To assess the coefficients of the plant $[g_1, g_2, g_3, g_4]$, the number of iterations was increased to 800, due to the higher number of dimensions of the search space in this case. The limits of each dimension of the search space are $[-10, 10]$.

In [12], the authors also demonstrate the robustness of the Passive System Identification attack in the face of sample loss. To evaluate such robustness, they considered four different rates l of sample loss: 0%, 5%, 10% and 20%. For each rate of sample loss, 100 different simulations were executed.

Figure 4 shows the mean estimated values of g_1, g_2, g_3, g_4, $c_{1,1}$ and $c_{2,1}$, considering the four mentioned rates of sample loss. All mean estimated values are represented with a Confidence Interval (CI) of 95%. The actual values of the coefficients of $C_1(z)$ and $G(z)$ are also depicted in Fig. 4. Additionally, the statistics (mean and standard deviation) of the estimated coefficients are presented in Table 2.

Regarding to the coefficients of $G(z)$, Fig. 4 shows that the difference between the mean and the actual values of g_1, g_2, g_3 and g_4 tends to raise with the increase of sample loss. It is also possible to note that the accuracy of the coefficients of $C_1(z)$ is better than the accuracy of the coefficients of $G(z)$, for all rates of sample loss. The means of $c_{1,1}$ and $c_{2,1}$ are closer to their actual values, with a smaller CI. In fact, the optimization process is more effective when computing the coefficients of $C_1(z)$ due to its smaller search space (which that has only two dimensions instead of the four dimensions of the $G(z)$ problem). In Fig. 4, it is possible to verify that, in all cases, the CIs tend to grow with the increase of the sample loss. The same thing occurs with the standard deviations shown in Table 2.

Despite the relative loss of accuracy of the Passive System Identification attack due to the increase of sample loss, such inaccuracy is not expressive even in the worst case (i.e. when $l = 20\%$). This behavior indicates the robustness of the Passive System Identification attack in the face of the loss of samples.

After estimating the models of the attacked plant and its respective control function, the next step is to design the data injection attack. In this sense, the authors of [12] designed an SD-Controlled Data Injection attack aiming to cause an *overshoot* of 50% in the rotational speed of the motor. As shown in Fig. 3, this SD-Controlled Data Injection attack is performed by a MitM in the forward stream. The attack was simulated in MATLAB, aiming to evaluate its accuracy when supported by the Passive System Identification attack.

The attack function executed by the MitM is $M(z) = \mathcal{K}_o$. Performing a root locus analysis considering the obtained models, the attacker adjusts $\mathcal{K}_o$ to make the system underdamped, with a peak of rotational speed 50% higher than its steady state speed. The values of

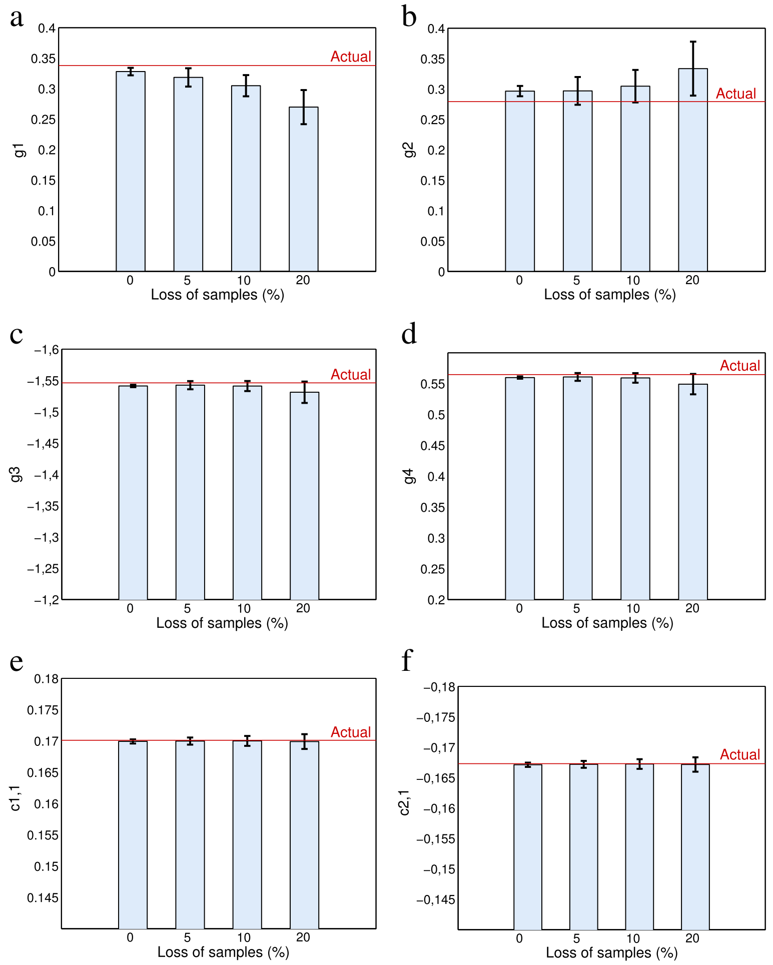

Fig. 4 Mean estimated coefficients of $G(z)$ and $C_1(z)$, in face of different rates of sample loss [12]. **a** g_1 of $G(z)$. **b** g_2 of $G(z)$. **c** g_3 of $G(z)$. **d** g_4 of $G(z)$. **e** $c_{1,1}$ of $C_{1(z)}$. **f** $c_{2,1}$ of $C_{1(z)}$

$\mathcal{K}_o$ were adjusted considering the mean estimated coefficients shown in Table 2. Table 3 shows the values of $\mathcal{K}_o$, estimated considering different rates of sample loss during the Passive System Identification attack, as well as the overshoots obtained with the respective $\mathcal{K}_o$ in the real model. In Fig. 5 it is possible to compare the response of the system without attack, with the response of the system with an attack aiming the overshoot of 50%. The curves referred as *estimated attack*, represent the results predicted by the attacker when the designed attack function $M(z)$ is applied to the estimated model – i.e. the model discovered by the attacker through to the Passive System Identification attack. On the other hand, the curves referred as *actual attack* represent the response of the actual system in the face of the same attack function $M(z)$. In other words, the curve *estimated attack* is the result achieved in a first moment, during the design stage of the attack, and the

Table 2 Statistics of the results obtained with different rates of sample loss [12]

Loss of	Mean						Standard deviation					
samples	g_1	g_2	g_3	g_4	$c_{1,1}$	$c_{2,1}$	g_1	g_2	g_3	g_4	$c_{1,1}$	$c_{2,1}$
0%	0.32793	0.29652	-1.54121	0.55983	0.16991	-0.16712	0.03097	0.04288	0.00986	0.00944	0.00167	0.00178
5%	0.31835	0.29689	-1.54251	0.56085	0.16997	-0.16719	0.07572	0.11523	0.03322	0.03194	0.00287	0.00287
10%	0.30473	0.30461	-1.54110	0.55925	0.16999	-0.16724	0.08781	0.13483	0.04076	0.03922	0.00397	0.00399
20%	0.26963	0.33352	-1.53119	0.54916	0.16989	-0.16716	0.14120	0.22378	0.08596	0.08313	0.00596	0.00598

Table 3 Values of $\mathcal{K}_o$ and overshoots obtained with the attacks [12]

	Sample loss in the passive system identification attack			
	0%	5%	10%	20%
$\mathcal{K}_o$	4.0451	4.0745	4.0828	3.796
Overshoot in the real model	48.90%	49.43%	49.57%	45.94%

curve *actual attack* is the result obtained in a second moment, when the designed attack is launched over the actual system. It is noteworthy that the attack to the actual model – represented by the *actual attack* curve – presents, in the time domain, a response quite similar to the attack estimated with the model obtained by the Passive System Identification attack – represented by the *estimated attack* curve. This can be verified not only in the case where the system is identified with 0% of sample loss, but also in the worst considered case, i.e. with 20% of sample loss. It is worth mentioning that all responses presented in Fig. 5 converge to the setpoint (1 rad/s).

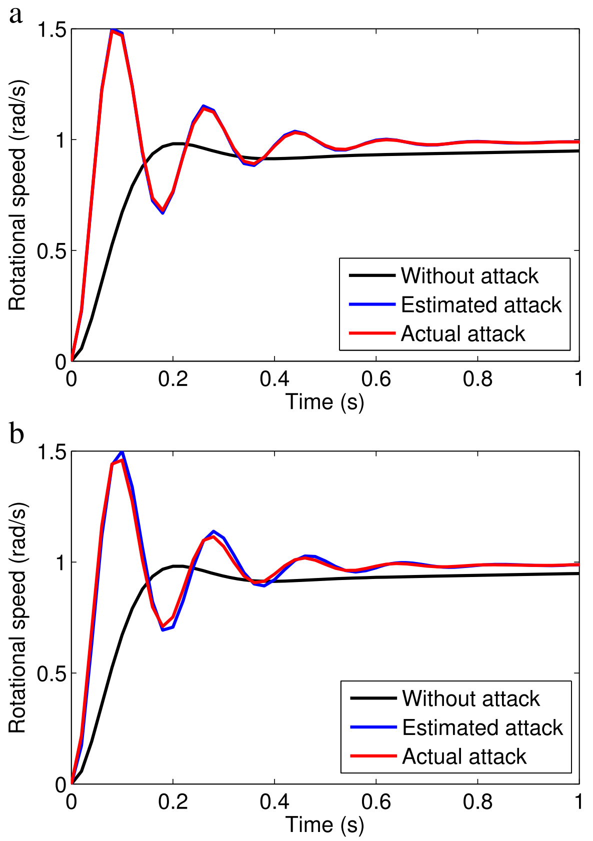

Fig. 5 Response of the plant to SD-Controlled Data Injection attacks designed to cause an overshoot of 50% in the rotational speed of the motor [12]. **a** Attack based on the data obtained without sample loss. **b** Attack based on the data obtained with 20% of sample loss

According to Table 3, it is possible to state that the SD-Controlled Data Injection attack, when supported by the Passive System Identification attack, is capable to accurately modify the physical response of the system, considering all evaluated rates of sample loss. In the worst case, i.e. with 20% of sample loss, it caused an overshoot of 45.94% (quite close to the desired 50%). Such accuracy allows the attacker to keep his offensive under control, leading the system to a behavior that is predefined as physically covert and capable to degrade the service performed by the plant under attack. These simulations provide conclusive data regarding the effectiveness of the Passive System Identification attack when it is used as a tool to support the design of a covert/model-dependent attack.

It is noteworthy that the manipulation of the rotational speed of a DC motor is used only to exemplify a physically covert interference in an NCS. This example is chosen due to the human difficulties to accurately estimate the rotation speed of objects under certain conditions. It is known, for instance, that under some conditions the apparent rotation speed is affected by the stimulus configuration (defined by the shape, size, and other characteristics of the rotating object) [28, 29]. Intuitively, it can be considered that, under those conditions, the perception of 50% of overshoot in the rotation speed may also be difficult to be perceived, especially because of its short duration. Although the authors of [12] use this example in their paper, it is worth mentioning that the concept of a physically covert attack can be extended to other interferences where, as defined in [12], the physical effects cannot be easily noticed or identified by a human observer, or can eventually be understood as a consequence of another cause, other than an attack.

4 Mitigation using switching controllers

As discussed in Section 1, one possible strategy to mitigate the Passive System Identification attack is to build the NCS with specific transfer functions that are harder to be identified. Therefore, it is necessary to analyze the two transfer functions $C(z)$ and $G(z)$, shown in Fig. 1, to verify what can be done to hinder the identification of the NCS. Regarding the plant, it is not desired or even feasible to modify its transfer function $G(z)$ just to make it harder to be identified. This follows from the simple fact that the plant's transfer function is a consequence of the physical structure of the controlled system. In other words, modify $G(z)$ means to modify the physical process being controlled, which is not convenient. However, it is reasonable to think about the design of controllers that are capable to meet, simultaneously, two objectives:

Objective I - Comply with the control requirements of the plant. In general, the primary requirement is to preserve the stability of the system. However, additional requirements – such as low settling time, low overshoot, etc. – may be considered depending on the process being controlled.

Objective II - Hinder the identification of the controller, so that the model obtained by the attacker is imprecise or ambiguous, in such a way that the attacker hesitates to launch covert or model-dependent attacks against the NCS.

Considering these two objectives, this work proposes the use of randomly switching controllers to mitigate Passive System Identification Attacks and, thus, prevent the design of covert/model-dependent attacks. Note that, the use of a switching controller does not avoid the identification of the plant's transfer function $G(z)$ by the Passive System Identification attack described in Section 3.1. Regardless of the controller switchings, the plant's transfer function is still an LTI system that can be identified by the mentioned System Identification attack, based on the analysis of the plant's input and output signals.

A Switching Controller, shown in Fig. 6, is composed by a set of N control functions $C_i(z)$, $i \in \mathcal{I} = \{1, \ldots, N\}$, that are switched by a switching rule S, to perform the control of a plant $G(z)$. If all control functions $C_i(z)$ and the plant's transfer function $G(z)$ are linear, as the NCS herein discussed, then the system is referred as a *switched linear system* (SLS). For the sake of clarity, but without loss of generality, in the present work, the switching controller is represented and discussed with only two control functions $C_1(z)$ and $C_2(z)$ – i.e. $N = 2$.

In a conventional switching controller [30–33], whose sole objective is to control the plant, the switching rule S, in general, orchestrates the switching events based on the plant and/or network behaviors. However, in the solution proposed in this work, the switching rule is not driven by the plant and/or network behaviors.

To achieve both Objectives I and II, the switching rule herein proposed operates as the Markov chain shown in Fig. 7. In this scheme, the control functions are switched at random intervals, in accordance with the probabilities $p_{11}(l)$, $p_{12}(l)$. $p_{21}(l)$ and $p_{12}(l)$, wherein l is the number of sampling intervals occurred since the last switch. The probabilities, $p_{12}(l)$ and $p_{21}(l)$ are taken from the probability density function (PDF) shown in Fig. 8, wherein a is the minimum number of sampling intervals that the system have to remain in the same state and b is the maximum number of sampling intervals that the system can remain in the same state. Note that $p_{11}(l) = 1 - p_{12}(l)$ and $p_{22}(l) = 1 - p_{21}(l)$.

The reason to switch at random intervals is that, according to [34], if the switching times are known, the identification of the SLS is straightforward. However, when the switching times are not available, the identification of the SLS turns into a nontrivial task. Moreover, even if the attacker obtain the plant's transfer function $G(z)$ and – somehow – discovers the control functions $C_i(z)$, the random switching rule still hinders the covert/model-dependent attack described in Section 3.2. This follows from the simple fact that it is more difficult to synchronize the interference caused by the covert/model-dependent attacks with the controller states, which are switched at random intervals.

However, despite the benefits that the switchings can bring from the point of view of a countermeasure, it can affect the stability of the NCS. Even if all subsystems of an SLS are stable, there are situations in which the switching events can make the SLS unstable. According to [7, 35], to be stable under arbitrary and unrestricted switchings, the SLS must meet two conditions:

1. All its subsystems must be asymptotically stable; and

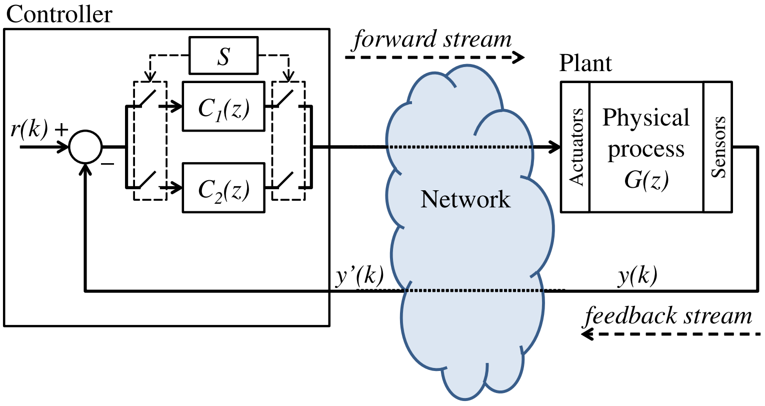

Fig. 6 NCS with a switching controller

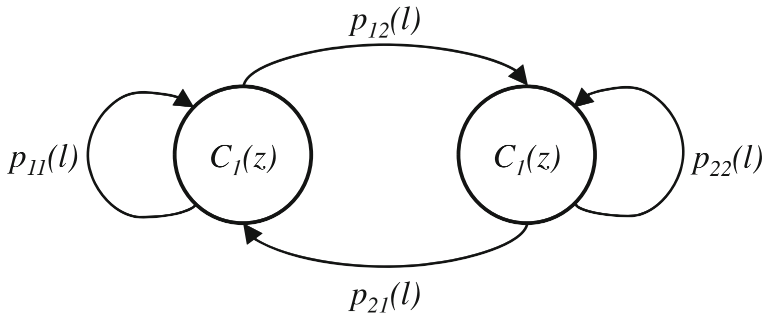

Fig. 7 Markov chain switching rule

2. There must exist a common Lyapunov function for all of its subsystems.

Note that, in the case of the NCS shown in Fig. 6, each subsystem is constituted by the plant transfer function $G(z)$ arranged in a closed loop with one control function $C_i(z)$. So, to make the NCS stable under arbitrary and unrestricted switching, all control functions $C_i(z)$, $i \in \mathcal{I} = \{1, 2\}$, have to be designed in order to meet the two aforementioned conditions.

Another valid strategy to obtain stability in an SLS with stable subsystems is by restricting the switching events. This can be done, for example, by establishing a minimum *dwell time* – i.e. the time between two consecutive switches. In an SLS, the instability generated when switching among two – or more – stable subsystems is caused by the failure to absorb the energy increase, caused by the switchings [35]. Intuitively, it is reasonable to think that if the SLS stays at stable subsystems long enough – using a slow switching rule – it becomes able to avoid the energy increase caused by the switchings, maintaining the desired stability. As proved in [36], it is always possible to preserve the stability of an SLS when all the subsystems are stable and the dwell time is sufficiently large. Actually, it is not critical if the SLS occasionally have a smaller dwell time, provided this does not occur too frequently. As demonstrated in [37], if all the subsystems are exponentially stable, then the SLS remains exponentially stable provided that the *average dwell time* is sufficiently large. In [38], this concept of *average dwell-time* is extended to the discrete-time switched systems – which is the case of an NCS endowed with the proposed countermeasure.

In the present work, instead of designing $C_1(z)$ and $C_2(z)$ to make the SLS stable under arbitrary and unrestricted switchings – i.e. meeting both conditions 1 and 2 – the restricted switching strategy is used. Thus, $C_1(z)$ and $C_2(z)$ are firstly designed based on the root-locus analysis, in order to make each subsystem stable. Then, the overall stability of the SLS is obtained by adjusting the parameters a and b of the PDF shown in Fig. 8, aiming an *average dwell-time* that makes the NCS stable.

Besides being adjusted for stability, parameters a and b also have to be adjusted to hinder the system identification attack. So, concerning Objective I, specifically for the sake of stability, a and b are increased as much as possible to ensure the minimum *average dwell-time* required for stability. On the other hand, concerning Objective II, a and b are adjusted to make the Passive System Identification Attack as much imprecise/ambiguous as possible, which not necessarily occur with high dwell

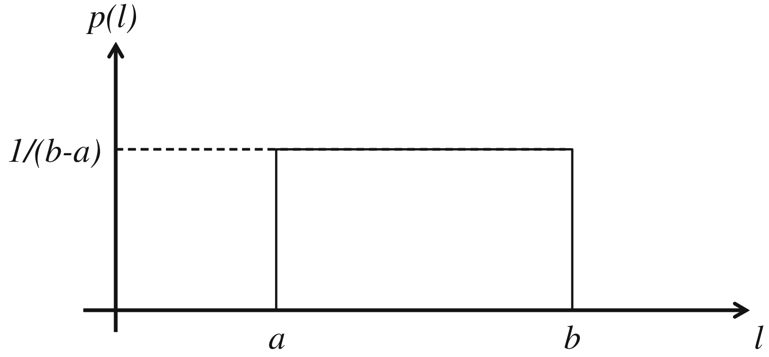

Fig. 8 PDF of p_{12} and p_{21}

times. In this sense, in this work, a and b are empirically adjusted in order to satisfy the two potentially conflicting objectives.

5 Results

As mentioned in Section 4, the design of the switching controller must meet simultaneously two objectives: hinder the identification process; and comply with the plant's control requirements. The results concerning these two objectives are presented in Sections 5.1 and 5.2, respectively, in order to demonstrate the feasibility of the solution from both perspectives. Additionally, Section 5.3 demonstrates the impact caused in the SD-Controlled Data Injection attack, described in Section 3.2, when the Passive System Identification Attack is mitigated by the proposed countermeasure.

In Sections 5.1 and 5.2, the results obtained with the proposed countermeasure are compared with the results obtained in an NCS without the proposed countermeasure – i.e. endowed with a non-switching controller. For this comparison, the NCS specified in Section 3.3 (with a non-switching controller) is used as reference.

The NCS with the proposed countermeasure has the same architecture shown in Fig. 6 and controls a DC motor whose transfer function is also defined by (6) – i.e. it controls the same plant that is controlled by the NCS with a non-switching controller described in Section 3.3. The sample rate of this system is also 50 samples/s and the set point $r(k)$ is a unitary step function. The switching controller has two control functions: $C_1(z)$, that is the same control function (5) of the non-switching controller; and $C_2(z)$ defined by (7),

$$C_2(z) = \frac{c_{1,2}z + c_{2,2}}{z - 1}. \tag{7}$$

wherein $c_{1,2} = 0.001$ and $c_{2,2} = 0.0002$. So, the NCS with the switching controller is an SLS with two subsystems. The control functions $C_1(z)$ and $C_2(z)$ are designed to make each subsystem stable – when separately analyzed – and are randomly switched based on the switching rule defined by the Markov chain and the PDF shown in Figs. 7 and 8, respectively. The parameters a and b of the PDF were empirically adjusted to $a = 40$ and $b = 60$, in order to meet Objectives I and II defined in Section 4. Regarding Objective I, it is worth mentioning that a and b were empirically adjusted aiming, primarily, the global stability of the SLS. However, the settling time and the overshoot of the plant are also evaluated in Section 5.2.

5.1 Mitigating the passive system identification attack

This section presents the results obtained by the Passive System Identification attack, when attacking both switching and non-switching controllers. For each controller, 100 attack simulations were performed. The parameters of the BSA are the same as those defined in Section 3.3, and the forward and feedback streams are also captured by the attacker during a period $T = 2s$ (100 samples). To evaluate the proposed countermeasure, we considered the scenario where the attacker obtained the best performance in Section 3.3 – i.e. without packet loss.

The coefficients estimated by all attack simulations (100 for each controller) are presented in Fig. 9. Recall that the non-switching controller just have one control function $C_1(z)$, while the switching controller has two control functions $C_1(z)$ and $C_2(z)$. Note that the actual values of the coefficients $[c_{1,1}, c_{2,1}]$ and $[c_{1,2}, c_{2,2}]$ of the two control functions $C_1(z)$ and $C_2(z)$, respectively, are also depicted in Fig. 9. By observing Fig. 9a and b, it is possible to state that the estimated coefficients of the non-switching controller are precise and accurate. In this case, the estimated coefficients are concentrated close to the actual values of $c_{1,1}$ and $c_{2,1}$. This concentration indicates that, with the non-switching controller, the

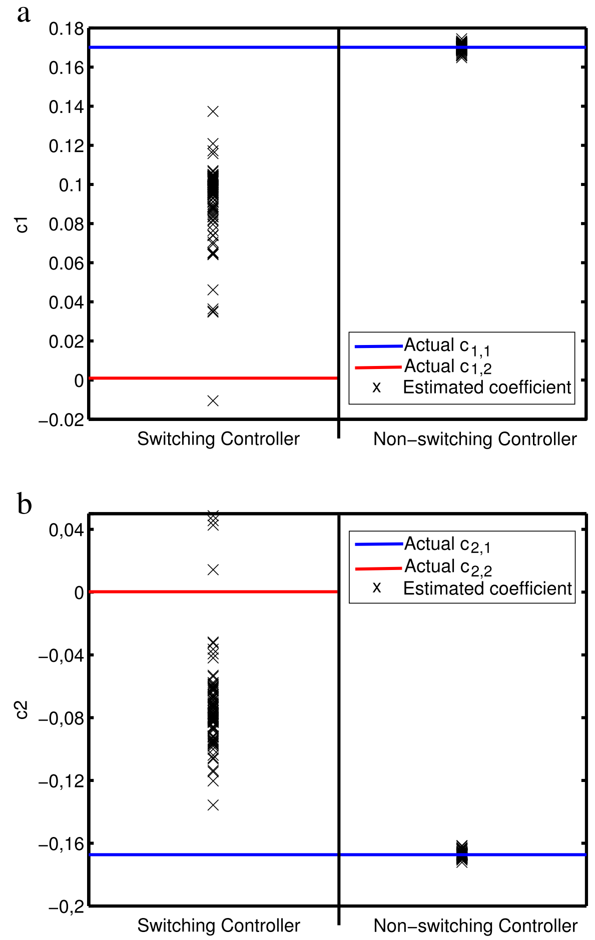

Fig. 9 Coefficients estimated by the passive system identification attack. **a** $c_{1,1}$ of $C_1(z)$ and $c_{1,2}$ of $C_2(z)$. **b** $c_{2,1}$ of $C_1(z)$ and $c_{2,2}$ of $C_2(z)$

Passive System Identification attack provides the information and the confidence that the attacker needs to design a covert/model-dependent attack – such as the SD-Controlled Data Injection attack demonstrated in Section 3.3. On the other hand, Fig. 9 shows that the use of the switching controller causes the dispersion of the estimated coefficients, reducing the precision and the accuracy of the Passive System Identification attack. With the switchings, the set of estimated values are spread and does not accurately indicate any of the coefficients of $C_1(z)$ and $C_2(z)$. It is worth mentioning that this spreading has a dissuasive effect. It increases the uncertainty of the attacker regarding the model of the attacked controller, in such way that the attacker may hesitate to proceed with his intention of a covert/model-dependent attack.

The impact of the switching controller in the attack performance can also be verified through the analysis of the global minimum values obtained for the fitness function (3). With the switching controller, the global minimum values of all attack simulations are between 2.64×10^{-04} and 8.53×10^{-04} (the mean is 7.42×10^{-04}, and the standard deviation is 1.70×10^{-04}). On the other hand, with the non-switching controller, all global minimum values are between 1.70×10^{-09} and 1.44×10^{-06} (the mean is 1.84×10^{-07}, and the standard deviation is 2.70×10^{-07}). Recall that, as discussed in Section 3.1.1, without sample loss, the minimum value of (3) is $\min f_j = 0$ when the attacked device is perfectly identified. So, the higher order of the global minimum values obtained with the switching controller also demonstrates the effectiveness of the proposed countermeasure. From the attacker point of view, these higher global minimum values may indicate that the Passive System Identification attack was not effective in obtaining the model of the attacked device. In this sense, the attacker must hesitate to launch covert/model-dependent attacks based on the information gathered by the Passive System Identification attack.

Another way to evaluate the impact of the proposed countermeasure in the Passive System Identification attack is through the zero-pole maps shown in Fig. 10. Figure 10a shows the zeros estimated by the simulations using the non-switching controller. Figure 10b, in turn, shows the zeros estimated by the simulations using the switching controller. Note that, in the simulations with the non-switching controller, the estimated zeros accurately meet the actual zero of $C_1(z)$. On the other hand, Fig. 10b shows that when the proposed countermeasure is used, the estimated zeros are spread and do not concur for the actual zeros of $C_1(z)$ and $C_2(z)$ – i.e. the control functions of the switching controller.

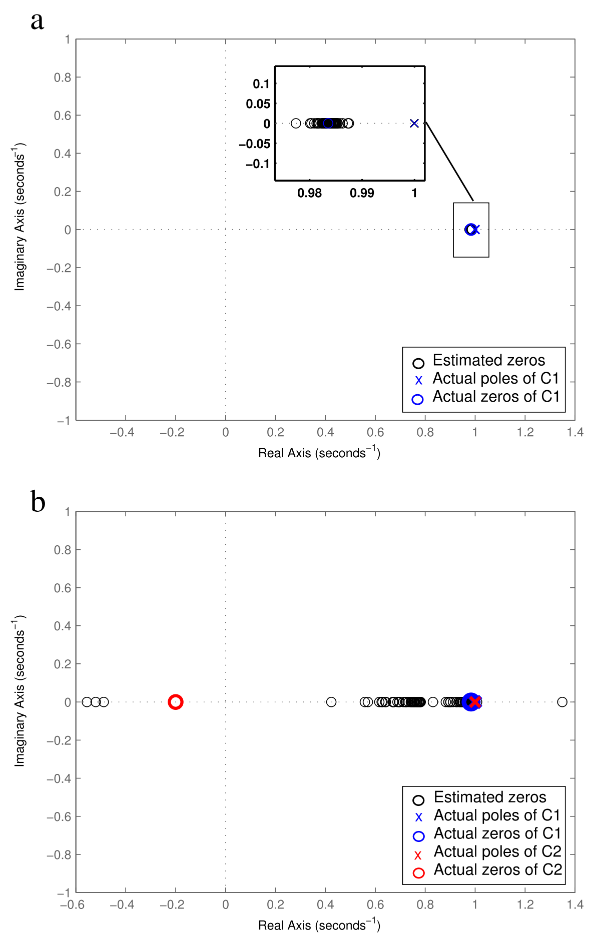

Fig. 10 Zeros and poles estimated by the Passive System Identification attack. **a** Using the non-switching controller. **b** Using the switching controller

It must be considered the possibility that the attacker, after some time, detects that the controller is changing its behavior over the time like a switching controller. In this case, it is reasonable to think that the attacker would try to estimate the control functions based on smaller monitoring periods T, to avoid measurements containing switching events. Considering this hypothesis, the performance of the Passive System Identification attack is evaluated using the following monitoring periods T: $0.2s$, $0.4s$, $0.6s$, $0.8s$, $1.0s$ and $1.2s$. Note that the maximum T in which the attacker can measure a signal without switchings is $T_b = 0.02b = 1.2s$. Therefore, to evaluate this tactic (of reducing T), the Passive System Identification attack is performed firstly during the execution of $C_1(z)$ and, after that, during the execution of $C_2(z)$. For the identification of $C_1(z)$ all monitoring periods start at $t = 0s$. For the identification of $C_2(z)$ all monitoring periods start at the first switching event (when $C_2(z)$ starts to be executed).

For each control function and each monitoring period, 33 attack simulations were executed. Figure 11 shows the

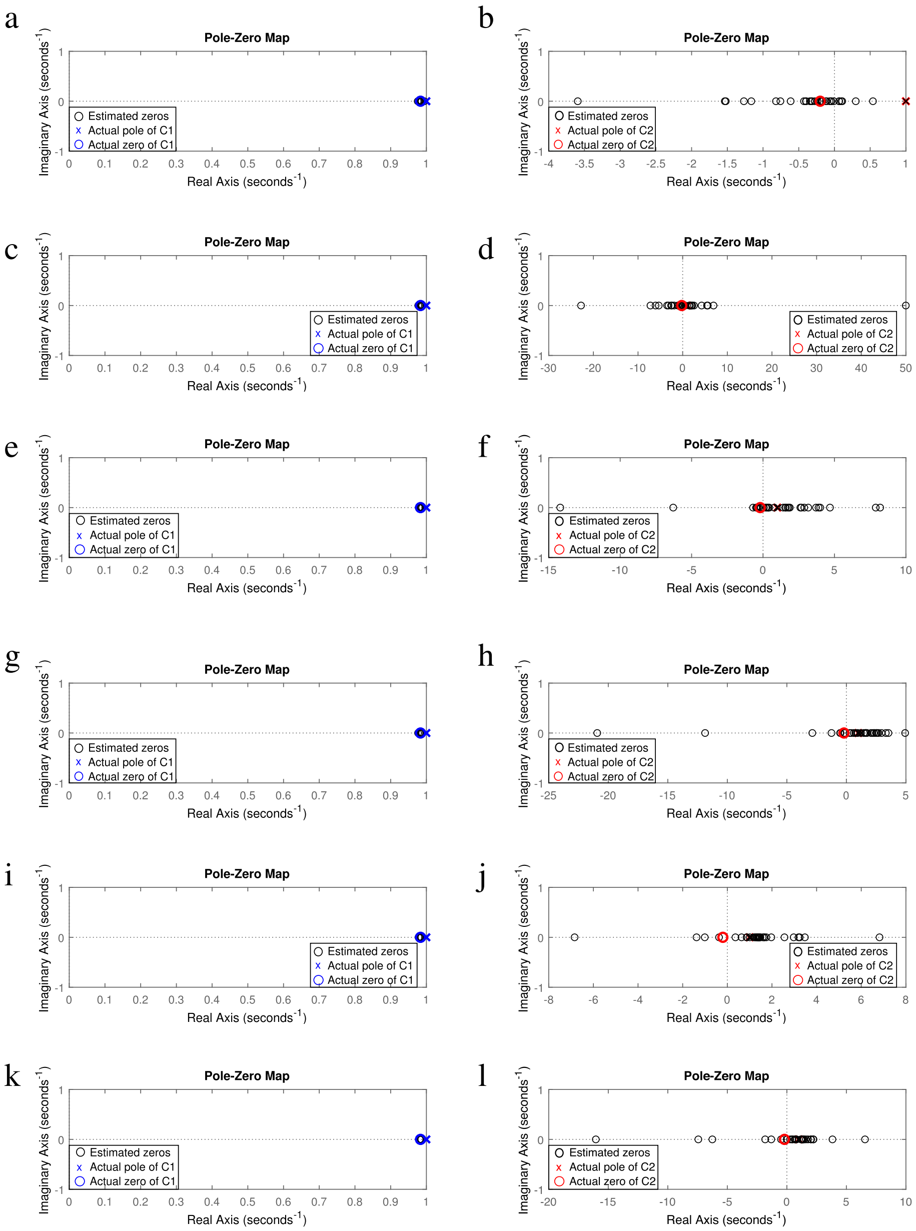

Fig. 11 Zeros and poles estimated by the Passive System Identification attack for smaller monitoring periods T (without a switching event during T). **a** Identifying C_1 with $T = 0.2s$ starting at $t = 0$. **b** Identifying C2 with $T = 0.2s$ starting at the first switching event. **c** Identifying C_1 with $T = 0.4s$ starting at $t = 0$. **d** Identifying C_2 with $T = 0.4s$ starting at the first switching event. **e** Identifying C_1 with $T = 0.6s$ starting at $t = 0$. **f** Identifying C_2 with $T = 0.6s$ starting at the first switching event. **g** Identifying C_1 with $T = 0.8s$ starting at $t = 0$. **h** Identifying C_2 with $T = 0.8s$ starting at the first switching event. **i** Identifying C_1 with $T = 1.0s$ starting at $t = 0$. **j** Identifying C_2 with $T = 1.0s$ starting at the first switching event. **k** Identifying C_1 with $T = 1.2s$ starting at $t = 0$. **l** Identifying C_2 with $T = 1.2s$ starting at the first switching event

estimated zeros of $C_1(z)$ and $C_2(z)$ considering each of the mentioned monitoring periods T. It is possible to verify that, for these monitoring periods, the estimated zeros of $C_1(z)$ are quite close to the actual zero. However, although $C_1(z)$ was satisfactorily identified with small T, Fig. 11 shows that, for all T, the estimated zeros of $C_2(z)$ are spread and do not accurately meet the actual zero of $C_2(z)$. These results indicate that small monitoring periods T may not be enough to identify some control functions, such as happened with $C_2(z)$. In this case, the switching controller arises as a good strategy to limit the available monitoring period, which causes difficulties for this metaheuristic-based Passive System Identification attack. Additionally, it is worth mentioning that even if the attacker somehow identifies all control functions $C_i(z)$, the random switching rule still mitigates the launch of a subsequent covert/model-dependent attack. As discussed in Section 4, this follows from the fact that it is more difficult to synchronize the interference caused by a covert/model-dependent attack with the controller states, which are switched at random intervals. Moreover, it is not trivial to find a single $M(z)$ capable to produce the intended controlled behavior for all $C_i(z)$ – in case the attacker choose this tactic to overcome the need to synchronize the covert/model-based attack.

The spreading of the estimated zeros in Fig. 10b, the inaccuracy of the estimated coefficients shown in Fig. 9, and the higher global minimum values found by the BSA demonstrate the effectiveness of the switching controllers in mitigating the Passive System Identification attack. With the proposed countermeasure, it is possible to state that the model obtained by the attacker is imprecise/ambiguous in such a way that the attacker may hesitate to launch a subsequent covert/model-dependent attack. Therefore, Objective II defined in Section 4 is met.

If an attacker, aiming to cause an overshoot of 50% in $y(k)$ (for example), implements an attack function $M(z)$ in the forward stream of an NCS, as shown in Fig. 3, then $y(k)$ is defined by (8):

$$y(k) = \mathcal{Z}^{-1}\left[\frac{C(z)M(z)G(z)}{1 + C(z)M(z)G(z)}R(z)\right]. \quad (8)$$

Similarly, if the attacker implements $M(z)$ in the feedback stream, then $y(k)$ is defined by (9):

$$y(k) = \mathcal{Z}^{-1}\left[\frac{C(z)G(z)}{1 + C(z)M(z)G(z)}R(z)\right]. \quad (9)$$

Note that in both cases, in the presence of an attack, the dynamics of $y(k)$ rely on $C(z)$, $G(z)$ and $M(z)$, considering that $R(z) = \mathcal{Z}\left[u(k)\right]$ is a step function. Therefore, if the attacker aims to cause an overshoot of 50% in $y(k)$, the design of $M(z)$ will require the knowledge of $C(z)$ and $G(z)$. The results shown in this section indicate that, with the proposed countermeasure, the attacker cannot accurately estimate the control functions of the NCS using the Passive System Identification attack. Therefore, even if the attacker is still able to identify the plant model (which is not mitigated by this countermeasure), he/she will not be able to design $M(z)$ to cause the 50% overshoot based only on the model of the plant, regardless of whether $M(z)$ is implemented in the forward or the feedback stream.

5.2 Complying the control requirements

In this section, the performance of the proposed countermeasure is analyzed from the control perspective. The aim of the simulations herein presented is to identify the possible impacts that the countermeasure may produce in the behavior of the plant. This analysis encompasses the following control aspects: stability; overshoot; and settling time. Considering these aspects, the performance of the switching controller is compared with the performance of the non-switching controller. Given the stochastic nature of the proposed countermeasure, which randomly switches among two control functions, the mentioned aspects are evaluated through a set of 100,000 simulations.

Figure 12 shows the responses of the plant, in the time domain, with and without the proposed countermeasure. The responses obtained with the proposed countermeasure – i.e. using the switching controller – are represented by the highlighted area. The bounds of this area are drawn based on the maximum and minimum values of the output $y(t)$ of the plant, considering all 100,000 simulations. In other words, when using the proposed countermeasure, all output signals $y(t)$ provided by the simulations are inside this area. The deterministic response of the plant without this countermeasure – *i.e.* when using the non-switching controller – is represented by the red line depicted in Fig. 12. Note that, for $0 \leq t \leq 0.8s$ the responses using the switching controller are the same as the response using the non-switching controller. This is caused by the minimum number of sampling intervals that the system has to remain in the same state, which is set to $a = 40$ samples (or $0.8s$, in the time domain).

Based on Fig. 12, considering all 100,000 simulations, it is possible to verify that the NCS with the proposed countermeasure is stable and the output of the plant does not present a stationary error – it always converges to the set point of 1 rad/s. In these aspects, from the control perspective, the proposed countermeasure presents the same performance as the non-switching controller. Also, the highlighted area indicates that the overshoots obtained with the countermeasure are not expressive, not exceeding 2.93% of the set point.

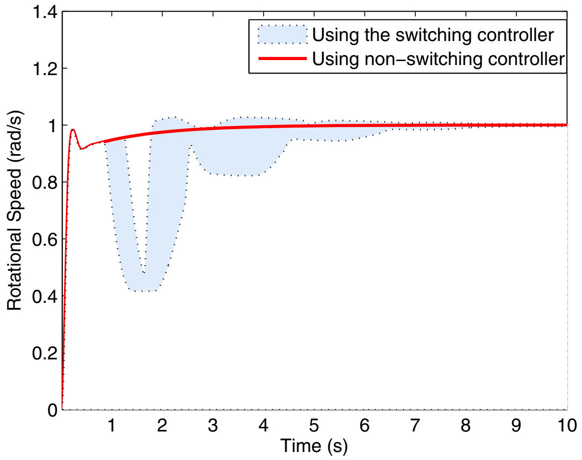

Fig. 12 Response of the plant in the time domain

However, due to the successive switchings, it is possible to verify in Fig. 12 that the settling time obtained with the proposed countermeasure is higher than the settling time obtained with the non-switching controller. With the non-switching controller, the deterministic settling time of the plant is 2.4s. On the other hand, with the switching controller, the settling time t_s of the plant is stochastic and depends on the random sequence of dwell times occurred before achieving t_s. The settling times of all 100,000 simulations using the switching controller are represented in the histogram shown in Fig. 13. The minimum and maximum settling times are 2.88s and 6.42s, respectively, and the mean is 4.2827s ± 0.0146s, with a confidence interval of 95%. It indicates that, regarding the settling time, the proposed countermeasure is less efficient than the non-switching controller.

It is worth mentioning that Fig. 12 exemplifies the behavior of the proposed countermeasure and compare its performance with the performance of an NCS with a non-switching controller. From this figure, it is possible to observe a behavioral profile that allows the evaluation of characteristics such as overshoot, settling time and stability. Regarding the latter, the stability of systems based on the average dwell time technique can be verified by the theory proposed in [38], which demonstrates the feasibility of the proposed countermeasure in terms of stability.

Note in Fig. 12 that the random switching rule adds to the system a variable (however, controlled and stable) behavior, which could reduce the ability of a human observer to detect slight manipulations caused by a physically covert attack. However, it is noteworthy that when an attacker designs a physically covert attack, as a premise, he/she does not aim to explore or manipulate physical behaviors that are easy to be noticed by a human observer. Instead of this, the attacker would manipulate physical behaviors that are not accurately perceived by a human observer. In this case, it is reasonable to consider that the variations caused by the switching controller will not significantly contribute for the poor perception of malicious and covert interferences that would naturally not be perceived by a human observer (even when a non-switching controller is used).

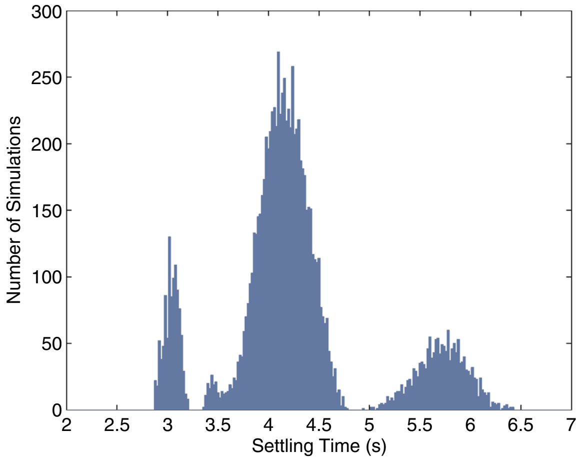

Fig. 13 Histogram of the settling time when using the proposed countermeasure

From the control perspective, the performance of the proposed countermeasure is satisfactory and, with the results presented in Section 5.1, indicates the feasibility of meeting both Objectives I and II, simultaneously. In the simulations of this section, the control provided by the switching controller presents a performance similar to the performance of the non-switching controller. The primary requirement of Objective I – i.e. stability – is met and the overshoots caused by the countermeasure, with the specified configurations, are not expressive. However, the simulations indicate an increase of the settling time of the plant, which may not be an issue, but have to be analyzed in the face of the specific process being controlled. In this sense, the results denote the existence of a tradeoff between hindering the identification attack and increasing the settling time of the system, which must be taken into account when deciding for using this countermeasure.

5.3 Impact in the controlled data injection attack

Consider that the attacker was not dissuaded by the uncertainties caused by the proposed countermeasure in the identification of the controller. Doing so, the aim of this section is to evaluate the impact of the proposed countermeasure in the design of an SD-Controlled Data injection attack.

The SD-Controlled Data Injection attack simulated in this section also aims to cause an overshoot of 50% in the rotational speed of the DC motor defined by (6), such as the attack described in Section 3.3. According to Section 3.2, to perform an SD-Controlled Data Injection attack, the attack function $M(z)$ must be designed based on the models of the plant and its controller.

As discussed in Section 4, the identification of the plant's transfer function $G(z)$ is not impacted by the use of the switching controller. So, the same $G(z)$ estimated in Section 3.3 (with a non-switching controller) is used in this section to design $M(z)$. Specifically, the coefficients used for $G(z)$ are the mean estimated coefficients shown in Table 2 for 0% of sample loss (which is the most accurate estimated model of $G(z)$). Regarding the model of the controller, as described in [12], $M(z)$ is designed considering the mean of the coefficients estimated for the switching controller. Then, performing a root locus analysis, the attacker designs the attack function (10), to make the system underdamped with a peak of rotational speed 50% higher than its steady state speed.

$$M(z) = 1.2815 \tag{10}$$

In Fig. 14, it is possible to compare the response that the attacker expects to obtain (referred as *Expected response*) with the responses that (10) actually produces (referred as *Actual responses*) when implemented in the real system. The *Expected response* represents what the attacker would obtain by simulating (10) in the forward stream of an NCS built with the models provided by the Passive System Identification attack. The *Actual responses* are represented by the highlighted area, whose bounds are drawn based on the maximum and minimum values of the output $y(t)$ of the plant, considering 100,000 simulations with (10) in the forward stream of the actual NCS. It means that, when (10) is implemented in the NCS all output signals $y(t)$ provided by the actual plant are inside this area.

It is worth mentioning that the aim of Fig. 14 is not to evaluate the stability of the proposed system after the execution of the SD-Controlled Data Injection attack (although in these simulations this system remained stable even after the execution of $M(z)$). The aim of Fig. 14 is to demonstrate that, with the proposed countermeasure, the interference produced by the attacker is not what he/she intended with the mentioned Data Injection attack. Note that, the actual responses of the plant are significantly different from the response that the attacker expects to obtain with the SD-Controlled Data Injection attack. These results are in contrast to the results achieved in the NCS with the non-switching controller, where the attack was accurate and executed exactly what was planned by the attacker, as shown in Section 3.3. With the proposed countermeasure, the maximum overshoot achieved by the plant was 10.12% (instead of the desired 50%). Notwithstanding, the highlight of these simulations is the fact that, with the proposed countermeasure, the information provided by the Passive System Identification attack is not useful to support the design covert/model-dependent attacks. This inaccurate information may lead the attacker to cause unpredictable results in the system, which may either be ineffective (not causing the desired degradation on the plant) or extreme (reducing the physical or cybernetic covertness of the attack). This analysis is consistent with the reasoning provided in Section 5.1. It demonstrates that when the NCS is endowed with the proposed countermeasure, the attacker must hesitate to launch a covert/model-dependent attack due to the inaccuracy of the Passive System Identification attack.

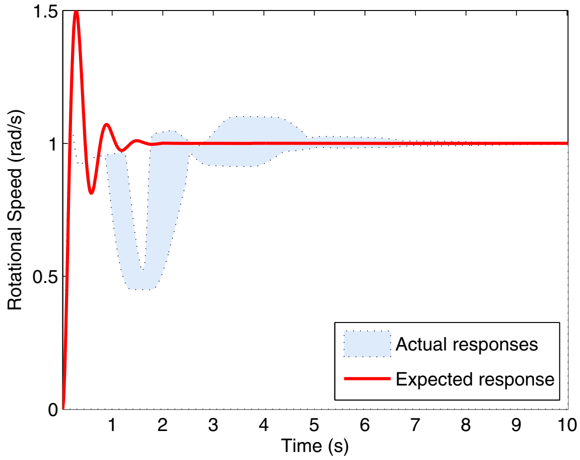

Fig. 14 Results of an SD-Controlled Data Injection attack in a system with the proposed countermeasure

Note that the countermeasure proposed in this paper aims to mitigate the Passive System Identifications attacks when the attacker is trying to obtain information about the control functions of the NCS. Consequently, it prevents the use of accurate information about these control functions in the design of a covert/model-dependent attack (such as a data injection attack in the forward stream of an NCS aiming to cause an overshoot or a steady state error). For instance, in an SD-Controlled Data Injection attack performed in the forward stream of the NCS, the attacker cannot cause a steady state error by just adding a step signal to $u(k)$, because the PI control functions will adjust the control signal to bring $y(k)$ back to 1 rad/s. Adding a ramp signal to $u(k)$ can cause a steady error in $y(k)$ for a while. However, it may not be a good strategy for the attacker, because at some time the controller and $u(k)$ will saturate, leading the plant to extreme behaviors (which is not desired if the attacker aims a physically covert attack). The alternative to cause a steady state error through the manipulation of the forward stream is to implement the attack function $M(z)$ exemplified in [12] which, to be designed, requires the knowledge about the controller and plant. Without the knowledge about the coefficients of the numerator of the PI control function, for example, the gain of $M(z)$ cannot be adjusted to cause the exact steady deviation of $y(k)$ that the attacker intends to cause. This makes the attack described in [12] model-dependent and, in this case, the countermeasure herein proposed is useful to hinder the attacker from obtaining the knowledge about the control functions of the NCS. On the other hand, in a system with an unitary feedback, it is possible to manipulate the steady state error of the plant by injecting data in the feedback stream, even when the attacker does not know the models of the plant and the controller. In this case, the manipulation of $y(k)$ can be interpreted as the direct manipulation of set point $r(k)$, which determines the steady state of the system. This attack, performed in the feedback stream is an example of data injection attack that is not model-dependent and, thus, should be mitigated by an additional countermeasure (complementary to the countermeasure proposed in this paper).

6 Conclusion

In this work, a randomly switching controller is proposed as a countermeasure for the Passive System Identification attack [12], in case of failure of other conventional security mechanisms – such as encryption, network segmentation and firewall policies. The simulations demonstrate that this countermeasure is capable to mitigate the mentioned attack, making the model obtained by the attacker imprecise and ambiguous. At the same time, the simulations demonstrate that the performance of the proposed countermeasure is satisfactory from the control perspective. Considering the control aspects, in general, the proposed countermeasure presents a performance similar to the performance of a non-switching controller, with an increase in the system's settling time. Therefore, when deciding for using this countermeasure, it must be considered the existence of a tradeoff between mitigate the identification attack and increase the settling time of the system – which, depending on the plant, is not necessarily a drawback.

As future work, we plan to evaluate the performance of this countermeasure when mitigating other system identification attacks/algorithms. Also, we encourage the development of a heuristic or an analytical method capable to provide control functions and switching rules that maximize the performance of the countermeasure in both mentioned objectives: comply with the plant's control requirements; and hinder the identification process.

Endnotes

[1] de Sa et al. [12] is an extended version of [13].

[2] The Passive System Identification attack was originally referred, in [12], as System Identification attack. However, with the introduction of the Active System Identification attack in [17], its designation was reviewed to Passive System Identification attack, in order to evince the differences between the two attacks.

Acknowledgements
This research was partially supported by the Brazilian research agencies CNPq and FAPERJ.

Authors' contributions
All authors contributed in all stages of this work, as well as read and approved the final manuscript.

Competing interests
The authors declare that they have no competing interests.

Publisher's Note

Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Author details
[1]Institute of Mathematics/NCE, Federal University of Rio de Janeiro, Av. Athos da Silveira Ramos, 274, 68.530 Rio de Janeiro, Brazil. [2]Admiral Wandenkolk Instruction Center, Brazilian Navy, Enxadas Island, Guanabara Bay, Rio de Janeiro, Brazil. [3]National Institute of Metrology, Quality and Technology, Av. Nossa Senhora das Graças, 50, Rio de Janeiro, Brazil. [4]Rio de Janeiro Federal Center for Technological Education, Av. Maracanã, 229, Rio de Janeiro, Brazil.

Received: 19 September 2017 Accepted: 6 December 2017
Published online: 05 February 2018

References

1. Tipsuwan Y, Chow MY, Vanijjirattikhan R. An implementation of a networked pi controller over ip network. In: Industrial Electronics Society, 2003. IECON'03. The 29th Annual Conference of the IEEE. Roanoke: IEEE. 2003. p. 2805–810.
2. Gupta RA, Chow MY. Networked control system: overview and research trends. Ind Electron IEEE Trans. 2010;57(7):2527–35.
3. Zhang L, Xie L, Li W, Wang Z. Security solutions for networked control systems based on des algorithm and improved grey prediction model. Int J Comput Netw Inf Secur. 2013;6(1):78.
4. Farooqui AA, Zaidi SSH, Memon AY, Qazi S. Cyber security backdrop: A scada testbed. In: Computing, Communications and IT Applications Conference (ComComAp). Beijing: IEEE. 2014. p. 98–103.
5. Chow MY, Tipsuwan Y. Network-based control systems: a tutorial. In: Industrial Electronics Society, 2001. IECON'01. The 27th Annual Conference of the IEEE. Denver: IEEE. 2001. p. 1593–1602.
6. Amin S, Litrico X, Sastry S, Bayen AM. Cyber security of water scada systems part i: analysis and experimentation of stealthy deception attacks. IEEE Trans Control Syst Technol. 2013;21(5):1963–70.
7. Dasgupta S, Routh A, Banerjee S, Agilageswari K, Balasubramanian R, Bhandarkar S, Chattopadhyay S, Kumar M, Gupta A. Networked control of a large pressurized heavy water reactor (phwr) with discrete proportional-integral-derivative (pid) controllers. IEEE Trans Nucl Sci. 2013;60(5):3879–88.
8. Ferrara A, Sacone S, Siri S. Model-based event-triggered control for freeway traffic systems. In: Event-based Control, Communication, and Signal Processing (EBCCSP), 2015 International Conference On. Krakow: IEEE. 2015. p. 1–6.
9. Singh R, Kuchhal P, Choudhury S, Gehlot A. Wireless controlled intelligent heating system using hpso. Procedia Comput Sci. 2015;48:600–5.
10. Xia YQ, Gao YL, Yan LP, Fu MY. Recent progress in networked control systems - a survey. Int J Autom Comput. 2015;12(4):343–67.
11. Smith RS. Covert misappropriation of networked control systems: Presenting a feedback structure. Control Syst IEEE. 2015;35(1):82–92.
12. de Sa AO, da Costa Carmo LFR, Machado RCS. Covert attacks in cyber-physical control systems. IEEE Trans Ind Inform. 2017;13(4):1641–51. doi:10.1109/TII.2017.2676005.
13. de Sa AO, da Costa Carmo LFR, Machado RCS. Ataques furtivos em sistemas de controle físicos cibernéticos. In: Anais do XVI Simpósio Brasileiro em Segurança da Informação e de Sistemas Computacionais (SBSeg). Niterói, Rio de Janeiro: SBC. 2016. p. 128–41.
14. Stouffer K, Pillitteri V, Lightman S, Abrams M, Hahn A. Nist special publication 800-82, revision 2: Guide to industrial control systems (ics) security. Gaithersburg: National Institute of Standards and Technology; 2015.
15. Pang ZH, Liu GP. Design and implementation of secure networked predictive control systems under deception attacks. IEEE Trans Control Syst Technol. 2012;20(5):1334–42.
16. Langner R. Stuxnet: Dissecting a cyberwarfare weapon. Secur Priv IEEE. 2011;9(3):49–51.
17. de Sa AO, da Costa Carmo LFR, Machado RCS. Bio-inspired active attack for identification of networked control systems. In: 10th EAI International Conference on Bio-inspired Information and Communications Technologies (BICT). New Jersey: ACM; 2017. p. 1–8. doi:10.4108/eai.22-3-2017.152407.
18. Long M, Wu CH, Hung JY. Denial of service attacks on network-based control systems: impact and mitigation. Ind Inform IEEE Trans. 2005;1(2): 85–96.
19. Snoeren AC, Partridge C, Sanchez LA, Jones CE, Tchakountio F, Schwartz B, Kent ST, Strayer WT. Single-packet ip traceback. IEEE/ACM Trans Networking (ToN). 2002;10(6):721–34.
20. Teixeira A, Shames I, Sandberg H, Johansson KH. A secure control framework for resource-limited adversaries. Automatica. 2015;51:135–48.

21. Smith R. A decoupled feedback structure for covertly appropriating networked control systems. In: Proceedings of the 18th IFAC World Congress 2011. Milano: IFAC-PapersOnLine; 2011.
22. Civicioglu P. Backtracking search optimization algorithm for numerical optimization problems. Appl Math Comput. 2013;219(15):8121–44.
23. Khatri S, Sharma P, Chaudhary P, Bijalwan A. A taxonomy of physical layer attacks in manet. Int J Comput Appl. 2015;117(22):6–11.
24. Zou Y, Wang G. Intercept behavior analysis of industrial wireless sensor networks in the presence of eavesdropping attack. IEEE Trans Ind Inform. 2016;12(2):780–7.
25. Stallings W. Cryptography and Network Security: Principles and Practices. New Jersey: Pearson Education India; 2006.
26. El-Sharkawi M, Huang C. Variable structure tracking of dc motor for high performance applications. Energy Convers IEEE Trans. 1989;4(4):643–50.
27. Tran T, Ha QP, Nguyen HT. Robust non-overshoot time responses using cascade sliding mode-pid control. J Adv Comput Intell Intel Inform. 2007;11:1224–1231.
28. Nishida S. Advancement of motion psychophysics: review 2001–2010. J Vis. 2011;11(5):11–11.
29. Blair CD, Goold J, Killebrew K, Caplovitz GP. Form features provide a cue to the angular velocity of rotating objects. J Exp Psychol Hum Percept Perform. 2014;40(1):116.
30. Skafidas E, Evans RJ, Savkin AV, Petersen IR. Stability results for switched controller systems. Automatica. 1999;35(4):553–64.
31. Liberzon D, Morse AS. Basic problems in stability and design of switched systems. IEEE Control Syst. 1999;19(5):59–70.
32. Safaei FRP, Ghiocel SG, Hespanha JP, Chow JH. Stability of an adaptive switched controller for power system oscillation damping using remote synchrophasor signals. In: Decision and Control (CDC), 2014 IEEE 53rd Annual Conference On. Los Angeles: IEEE. 2014. p. 1695–1700.
33. Ferrara A, Sacone S, Siri S. A switched ramp-metering controller for freeway traffic systems. IFAC-PapersOnLine. 2015;48(27):105–10.
34. Wang J. Identification of switched linear systems. PhD thesis. 2013.
35. Lin H, Antsaklis PJ. Stability and stabilizability of switched linear systems: a survey of recent results. IEEE Trans Autom Control. 2009;54(2):308–22.
36. Morse AS. Supervisory control of families of linear set-point controllers-part i. exact matching. IEEE Trans Autom Control. 1996;41(10): 1413–31.
37. Hespanha JP, Morse AS. Stability of switched systems with average dwell-time. In: Decision and Control, 1999. Proceedings of the 38th IEEE Conference On. Phoenix: IEEE. 1999. p. 2655–660.
38. Zhai G, Hu B, Yasuda K, Michel AN. Qualitative analysis of discrete-time switched systems. In: American Control Conference, 2002. Proceedings of the 2002. Anchorage: IEEE. 2002. p. 1880–1885.