Bio-inspired Active Attack for Identification of Networked Control Systems

Alan Oliveira de Sá
Institute of Mathematics/NCE,
Federal University of
Rio de Janeiro
ZIP Code: 21.941-901
CIAW, Brazilian Navy
Rio de Janeiro – RJ – Brazil
alan.oliveira.sa@gmail.com

Luiz F. R. da C. Carmo
Institute of Mathematics/NCE,
Federal University of
Rio de Janeiro
ZIP Code: 21.941-901
National Institute of Metrology,
Quality and Technology
Duque de Caxias, RJ, Brazil
lfrust@inmetro.gov.br

Raphael C. S. Machado
National Institute of Metrology,
Quality and Technology
Duque de Caxias, RJ, Brazil
rcmachado@inmetro.gov.br

ABSTRACT

The use of communication networks to interconnect controllers and physical plants in industrial and critical infrastructure facilities exposes such control systems to threats typical of the cyber domain. In this sense, studies have been done to explore vulnerabilities and propose security solutions for Networked Control System (NCS). From the point of view of the control theory, the literature indicates that stealthy and accurate cyber-physical attacks must be planned based on an accurate knowledge about the model of the NCS. However, most literature about these attacks does not indicate how such knowledge is obtained by the attacker. So, to fill this hiatus, it is proposed and evaluated in this paper an Active System Identification attack, where the attacker injects data on the NCS to learn about its model. The attack is implemented based on two bio-inspired metaheuristics, namely: Backtracking Search Optimization Algorithm (BSA); and Particle Swarm Optimization (PSO). The results indicate a better performance of the BSA-based attack, especially when the captured signals contain white Gaussian noise. The goal of this paper is to demonstrate the degree of accuracy that this attack may achieve, highlighting the potential impacts and encouraging the research of possible countermeasures.

CCS Concepts

•**Security and privacy** → **Formal security models;** Cryptanalysis and other attacks; •**Computing methodologies** → **Search methodologies; Computational control theory;**

Keywords

Security, Cyber-Physical Systems, Networked Control Systems, System Identification, Backtracking Search Algorithm, Particle Swarm Optimization

BICT 2017, March 15-16, Hoboken, United States

1. INTRODUCTION

System identification, *i.e.* the action of building mathematical models of dynamic systems, is often used to obtain the model of physical processes aiming to subsidize the design of their respective control systems. However, it can also be considered a key step for the execution of stealth – or covert, as mentioned in [16, 17, 20] – attacks against Networked Control Systems (NCS). Indeed, to reduce the probability to be detected by algorithms that monitor the dynamics of the controlled plant, the attacker must have an accurate model of the targeted system, such as demonstrated in [16, 17, 20].

A possible strategy to obtain information about the model of the targeted system is through passive System Identification attacks, as reported in [5]. In this technique, the attacker eavesdrops the communications between the controller, actuators and sensors of the NCS until enough information is collected to determine the parameters of the plant and its control system. Such passive approach can make the system identification to last for a long time, until meaningful information transits at the eavesdropped communication line. The situation is even worse if the system is on steady state, because no meaningful information may transit through the NCS's communication links for a long time – indeed, the information content of the signals measured under steady operating conditions is often insufficient for identification purposes [22]. This attacker's constraint may be overcome by Active System Identification attacks, which, as far as we know, is not reported in the literature.

In this sense, in the present work, we propose an active attack for the identification of NCSs. Our approach was inspired by the classic active cryptanalytic attacks – chosen plaintext and chosen cypher text –, where the attacker inserts messages in the crypto-engine, in opposition to passive attacks – cyphertext-only, known plaintext –, where the attacker simply listen the communication channels and passively collects information [19].

In the attack herein proposed, a specially tailored signal is inserted by the attacker in an NCS communication channel and, by observing the behavior of the system in closed-loop, the attacker determines the parameters of its open-loop transfer function. To do so, the attacker just needs to intercept one communication channel of the NCS, where the attacker both insert the attack signal and listen the conse-

quent system response. The knowledge of the NCS's open-loop transfer function, obtained through this attack, is useful for the design of other sophisticated attacks. For instance, if an attacker learns the open-loop transfer function of an NCS, it is possible to further design attacks capable to accurately change the transient response and/or steady state response of the plant, such as demonstrated in [5], causing, for example, stationary errors or overshoots on the plant. A stationary error may reduce the efficiency of the physical process, while overshoots may cause stress and possibly damages [6, 21] to the plant, reducing its mean time between failure (MTBF).

The present Active System Identification attack is developed based on two bio-inspired metaheuristics, whose results are analyzed and compared, namely: the Backtracking Search Optimization algorithm (BSA) [4]; and the Particle Swarm Optimization (PSO) [10]. If the attack signal $a(k)$ and the consequent response $y_a(k)$ of an NCS is known, its open-loop transfer function can be assessed by applying $a(k)$ in an estimated model, which is adjusted until its estimated output $\hat{y}_a(k)$ matches $y_a(k)$. In this sense, the BSA and the PSO are used to iteratively adjust the parameters of an estimated model, by minimizing a specific fitness function, until the estimated model converges to the actual model of the NCS. The BSA and the PSO are chosen to perform this task due to their capability to converge to good solutions, such as demonstrated in [9, 13, 23, 24, 8] specifically for control system problems.

It is worth mentioning that the Active System Identification attack herein proposed is different from the active attacks performed to identify vulnerabilities of protocols and applications within the layers of the OSI model, such as the active scanning process used to identify network services [2].The attack herein proposed aims to identify the physical model of a plant that, in an NCS, lies above the application layer of the OSI model.

The goal of this paper is to demonstrate the degree of accuracy that such attack may achieve, highlighting its potential impacts and encouraging the research of countermeasures capable to prevent or detect the execution of this kind of attack. The remainder of this paper is organized as follows. In Section 2, we review the literature on NCS attacks, with focus on the intelligence gathered to subsidize their design. In Sections 3 and 4, there are provided brief descriptions of the BSA and PSO, respectively. In Section 5, it is described the Active System Identification attack, herein proposed. In Section 6, there are presented and compared the results achieved by the proposed attack, using both metaheuristics, in simulations where the NCS is constituted by a DC motor and a proportional-integral (PI) controller. Section 7 contains our final considerations.

2. RELATED WORKS

The possibility of large impact cyber-physical attacks became unprecedentedly concrete after the launch of the Stuxnet worm [11] and has been motivating researches concerning the security of NCSs. In this section, it is presented a review of the literature related to this subject.

In [12] the authors propose two queueing models that are used to evaluate the impact of delay jitter and packet loss in an NCS under attack. The attack is not designed taking into account the models of the controller and the physical plant. Such models are unknown by the attacker. Thus, to affect the plant's behavior, the attacker arbitrarily floods the network with traffic, causing jitter and packet loss. In this method of attack, the excess of packets in the network can reduce the stealthiness of the attack, allowing the adoption of countermeasures, such as packet filtering [12] or blocking the malicious traffic on its origin [18]. Moreover, 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 stealth attack is intended.

In [7], it is presented a testbed for Supervisory Control and Data Acquisition (SCADA) using TrueTime – a MATLAB/Simulink based tool. The authors demonstrate an attack where a malicious agent transmits false signals to the controller and actuator of an NCS. The false signals are randomly generated, aiming to make a DC motor lose its stability. This kind of attack does not require a previous knowledge about the plant and controller of the NCS. The drawback is that the desired physical effect and the stealthiness of the attack can not be ensured due to the unpredictable consequences of the application of random false signals to a system which the model is not known.

A general framework for the analysis of a wide variety of attacks over NCSs is provided in [20]. The authors classify and establish the requirements for the attacks in terms of the model knowledge, disclosure and disruption resources. In their work, it is stated that covert attacks require high level of knowledge about the model of the targeted system. Exemples of covert attacks that agree with this statement are provided in [16, 17]. In these works the attacks are performed by a man-in-the-middle (MitM), where the attacker needs to know the model of the plant under attack and also inject false data in both the forward and the feedback streams. The stealthiness of the attacks described in [16, 17] is analyzed from the perspective of the signals arriving to the controller, and depends on the difference between the actual model of the plant and the model known by the attacker. In [1], it is demonstrated another stealth attack where the attacker, aware of the system's model, injects an attack signal in the NCS to steal water from the Gignac canal system located in Southern France.

Table 1: Synthesis of the related attacks

Attack	Method	System knowledge	How the knowledge is obtained
Stuxnet *worm* [11]	Modifications in the PLC code	Yes	Experiments in a real system
Long, *et al.* [12]	Inducing *jitter* and packet loss	None	N/A
Farooqui, *et al.* [7]	Data injection	None	N/A
Smith [16, 17]	Data injection	Yes	Not described
Teixeira [20]	Packet loss	None	N/A
	Data injection	Yes	Not described
Amin [1]	Data injection	Yes	Not described
SD-Controlled [5]	Data injection	Yes	Passive system identification

In [1, 16, 17, 20], where it is required a previous knowledge about the models of the NCS under attack, it is not des-

cribed how this knowledge is obtained by the attacker. It is just stated that a model is previously known to subsidize the design of the attack. More recently, in [5], the authors propose a System Identification attack to fill this hiatus. They demonstrate how the data required for the design of Denial-of-Service (DoS) or Service Degradation (SD) attacks may be obtained through a passive System Identification attack. The attack proposed in [5] does not need to inject signals on the NCS to estimate its models. However, it depends on the occurrence of events, that are not controlled by the attacker, to produce signals that carry meaningful information for the system identification algorithm. The Active System Identification attack herein proposed, constitutes an alternative to the passive System Identification attacks in situations where the attacker may not wait so long for the occurrence o such meaningful signals. A synthesis of the characteristics of the attacks referred in this section is presented in Table 1.

3. BACKTRACKING SEARCH ALGORITHM

In this section, there are described the basic concepts of the BSA, in order to provide a clear comprehension regarding to 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 with [4], 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 composes the BSA's memory.

During the first selection stage (Selection-I), the algorithm randomly determines, based on an 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, it shuffles the individuals of this population.

The mutation operator creates $\mathcal{P}_{mod}$, which is the preliminary version of the new population $\mathcal{P}_{new}$). It does so according to (1):

$$\mathcal{P}_{mod} = \mathcal{P} + \eta \cdot \Gamma(\mathcal{P}_{hist} - \mathcal{P}), \tag{1}$$

wherein η is empirically adjusted through simulations and $\Gamma \sim N(0,1)$, with 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 $(\mathcal{P}_{hist} - \mathcal{P})$.

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

At the second selection stage (Selection-II), the algorithm evaluates, selects elements of $\mathcal{P}_{new}$ (*i.e.* individuals obtained after mutation and crossover), which should have better fitness than those in $\mathcal{P}$ (*i.e.* individuals before applying both the operators of crossover and mutation) and replaces them in $\mathcal{P}$. Hence, $\mathcal{P}$ includes only new individuals that should have 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 both good exploration and exploitation capabilities. With this parameters set, the BSA is used to search for the global minimum of the fitness function described in Section 5.

4. PARTICLE SWARM OPTIMIZATION

PSO has roots in the collective behavior of social models such as bird flocking and fish schooling. A particle, *i.e.* the basic element of the algorithm, represents a possible solution of a problem. Thus, the swarm represents a set of possible solutions. At each iterative cycle, the position of each particle is updated according to (2), where x_j and v_j are the position and velocity of particle j, respectively.

$$x_j(t+1) = x_j(t) + v_j(t+1) \tag{2}$$

The computation of v_j considers three terms: the particle's inertia; the particle's cognition, which is based on the best solution found by the particle so far; and social term, which is based on global best solution found by the swarm. The velocity of particle j, at each dimension d, is defined in (3):

$$\begin{aligned} v_{jd}(t+1) = \omega v_{jd}(t) + \varphi_1 r_{1d}(t)(m_{jd} - x_{jd}(t)) \\ + \varphi_2 r_{2d}(t)(m_{gd} - x_{jd}(t)), \end{aligned} \tag{3}$$

wherein ω is a parameter that weighs the inertia of the particle, φ_1 and φ_2 are parameters that weigh the cognitive and social terms, respectively, r_1 and r_2 are random numbers in [0,1], m_j is the best position visited by particle j so far, and m_g is the best position discovered by the swarm considering the experience of all the particles.

In order to better explore multi-dimensional search spaces, a velocity limit is imposed for each dimension d, as in (4):

$$0 \leq v_{jd} \leq \delta(max_d - min_d), \tag{4}$$

wherein max_d and min_d are the maximum and minimum limits of the search space at each dimension d and $\delta \in [0,1]$.

The overall computation that the PSO performs to minimize a fitness function $f(x)$ is given in Algorithm 2, where x is the particle position and S is the swarm size.

Algorithm 2 PSO Algorithm

begin
  for *each particle* j, $1 \leq j \leq S$ **do**
    Set randomly position x_j and velocity v_j;
    $m_j \leftarrow x_j$;
  end
  $m_g \leftarrow$ *smallest* $m_j, 1 \leq j \leq S$;
  repeat
    for *each particle* j, $1 \leq j \leq S$ **do**
      Update velocity v_j, as in (3) and (4);
      Update position x_j, as in (2);
      $fitness \leftarrow f(x_j)$;
      $m_k \leftarrow x_j$, whenever $fitness < f(m_j)$;
      $m_g \leftarrow x_j$, whenever $fitness < f(m_g)$;
    end
  until *Stopping condition*;
  return m_g;
end

5. THE ACTIVE SYSTEM IDENTIFICATION ATTACK

The Active System Identification attack, herein proposed, is intended to assess the coefficients of a transfer function $G(z) = C(z)P(z)$ of an NCS, wherein $C(z)$ is the controller's control function and $P(z)$ is the plant's transfer function as shown in Figure 1. The transfer functions are all linear time-invariant (LTI). This attack is performed by a MitM that may be located either in the forward or in the feedback link. For the sake of clarity of the analysis presentation, but without loss of generality, we focus on the case where the MitM is in the feedback link, *i.e.* between the plant's sensors and the controller's input. To estimate the model of the attacked NCS, the attacker injects an attack signal $a(k)$, and measure the response of the system to such signal.

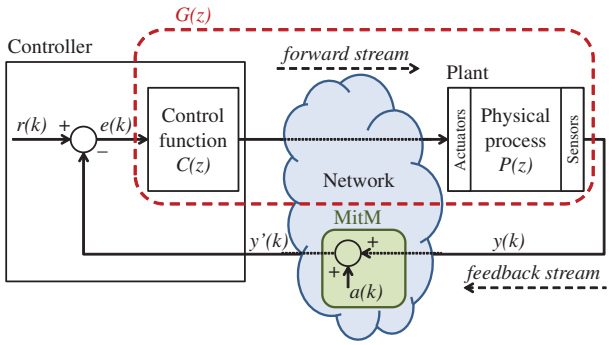


Figure 1: Active System Identification attack with a MitM in the feedback link.

The complete response of the generic NCS shown in Figure 1, considering only the inputs $R(z) = \mathcal{Z}[r(k)]$ and $A(z) = \mathcal{Z}[a(k)]$, is expressed in the z domain by (5):

$$Y(z) = \frac{G(z)}{1+G(z)}R(z) - \frac{G(z)}{1+G(z)}A(z), \tag{5}$$

wherein $Y(z) = \mathcal{Z}[y(k)]$. $\mathcal{Z}$ represents the Z-transform operation. As a premise, in a normal condition, it is considered that $a(k) = 0$ and the system is designed to make $y(k) \to q$, in such way that $y(k) \approx q \ \forall k > k_s$, *i.e.* the output $y(k)$ of the NCS converges and stabilizes at a constant value q after a certain amount of samples k_s. Indeed, it is usually one of the main aims of a control system. Now, considering $a(k) \neq 0$, the output $y(k)$, $\forall k > k_s$, may be defined approximately as (6):

$$y(k) = q - \mathcal{Z}^{-1}\left[\frac{G(z)}{1+G(z)}A(z)\right], \forall k > k_s. \tag{6}$$

Thus, after k_s, the portion of $y(k)$ caused by $r(k)$ can be eliminated by just subtracting q from (6), which leads to (7):

$$y_a(k) = y(k) - q = -\mathcal{Z}^{-1}\left[\frac{G(z)}{1+G(z)}A(z)\right], \forall k > k_s. \tag{7}$$

wherein $y_a(k)$ represents the portion of $y(k)$ caused by the attack signal $a(k)$. The value of q can be assessed by the attacker through an eavesdropping attack in the feedback stream, by just capturing $y(k)$ after the stabilization of the NCS. The subtraction of q after k_s makes the system identification attack independent of $r(k)$ $\forall k > k_s$. The Active System Identification attack now just relies on the attack signal $a(k)$, which can be chosen, and the response of the system to the attack $y_a(k)$ can be obtained in accordance with (7). The signal $y_a(k)$ starts with $a(k)$ and has the size of a monitoring period T.

If the attack input $a(k)$ and its consequent output $y_a(k)$ are known, the model of $G(z)$ can be assessed by applying the known $a(k)$ in an estimated system, defined by (8):

$$\hat{y}_a(k) = -\mathcal{Z}^{-1}\left[\frac{G_e(z)}{1+G_e(z)}\right] * a(k), \tag{8}$$

wherein $G_e(z)$ is the estimation of $G(z)$ and $\hat{y}_a(k)$ is the output of the estimated system in face of $G_e(z)$. By comparing $\hat{y}_a(k)$ with $y_a(k)$, the attacker is capable to evaluate whether $G_e(z)$ is equal/approximately $G(z)$. Note that $G_e(z)$ is a generic transfer function represented by (9):

$$G_e(z) = \frac{\alpha_n z^n + \alpha_{n-1}z^{n-1} + ... + \alpha_1 z^1 + \alpha_0}{z^m + \beta_{m-1}z^{m-1} + ... + \beta_1 z^1 + \beta_0}, \tag{9}$$

wherein n and m are the order of the numerator and the denominator, respectively, and $[\alpha_n, \alpha_{n-1}, ...\alpha_1, \alpha_0]$ and $[\beta_{m-1}, \beta_{m-2}, ...\beta_1, \beta_0]$ are the coefficients of the numerator and the denominator, respectively, that are intended to be found by this Active System Identification attack. Thus, to find $G(z)$, the coefficients of $G_e(z)$ are adjusted until the estimated output $\hat{y}_a(k)$ converges to the known $y_a(k)$.

In this sense, the BSA and the PSO are used to iteratively adjust the estimated model, by minimizing a specific fitness function presented in this section, until the estimated model $G_e(z)$ converges to the actual $G(z)$ of the real NCS. To compute the fitness of the individuals of the optimization algorithm, *i.e.* the BSA or PSO, the same attack signal $a(k)$ that provided $y_a(k)$, according with (7), is applied on the estimated system defined by (8) and (9), where the coefficients of $G_e(z)$ are the coordinates $x_j = [\alpha_{n,j}, \alpha_{n-1,j}, ...\alpha_{1,j}, \alpha_{0,j}, \beta_{m-1,j}, \ \beta_{m-2,j}, ...\beta_{1,j}, \beta_{0,j}]$ of an individual j of the BSA/PSO. The output $\hat{y}_{aj}(k)$ is the response of the estimated model (8) (9), in face of $a(k)$, when the coefficients

of $G_e(z)$ are x_j. So, the fitness f_j of each individual j is obtained comparing $\hat{y}_{aj}(k)$ with $y_a(k)$, according with (10):

$$f_j = \frac{\sum_{k=0}^{N} (y_a(k) - \hat{y}_{aj}(k))^2}{N}, \quad (10)$$

wherein N is the number of samples that exist during the monitoring period T of $y_a(k)$. Note that, if no other inputs – perturbation or noise – occur in the NCS during T, then $\min f_j = 0$ when $[\alpha_{n,j}, \alpha_{n-1,j}, ...\alpha_{1,j}, \alpha_{0,j}, \beta_{m-1,j}, \beta_{m-2,j}, ... \beta_{1,j}, \beta_{0,j}] = [\alpha_n, \alpha_{n-1}, ...\alpha_1, \alpha_0, \beta_{m-1}, \beta_{m-2}, ...\beta_1, \beta_0]$, *i.e.* when the estimated $G_e(z)$ converges to $G(z)$.

An analogy may be established between this Active System Identification attack and the Chosen Plaintext cryptanalytic attack [19], wherein $a(k)$ corresponds to the chosen plaintext, $y_a(k)$ represents the ciphertext, the equations (8) and (9) together correspond to the encryption algorithm and the actual coefficients $[\alpha_n, \alpha_{n-1}, ...\alpha_1, \alpha_0]$ and $[\beta_{m-1}, \beta_{m-2}, ...\beta_1, \beta_0]$ of $G_e(z)$ correspond to the secret key.

6. RESULTS

In this section, there are presented and analyzed the results obtained with simulations of the proposed Active System Identification attack. The attacked system, shown in Figure 2, consists of a DC motor whose rotational speed is controlled by a Proportional-Integral (PI) controller. This example is chosen due to the use of DC motors in a vast number of real world control systems. Moreover, DC motors has been widely used in previous works about NCS [3, 12, 14, 15]. It is noteworthy that the model herein chosen as an example does not exhaust the potential targets for this attack. NCSs composed by another kinds of LTI devices may also be a target. However, it must be taken into account that the computational cost of the attack, when launched over different LTI systems, may vary with the number of their unknown coefficients – *i.e.* the number of dimensions of the search space explored by the optimization algorithms (BSA or PSO, in this paper).

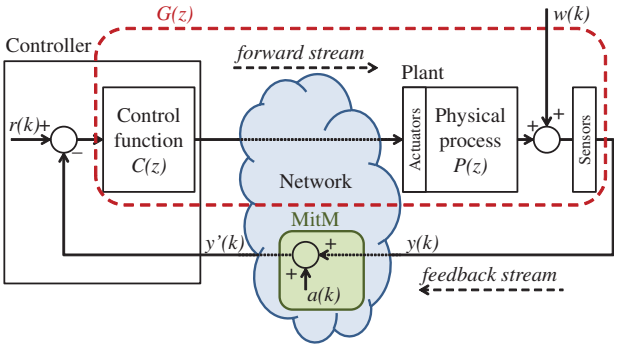


Figure 2: Active System Identification attack on noisy NCS.

The PI control function $C(z)$ and the DC motor transfer function $P(z)$, obtained from [12], are represented by (11):

$$C(z) = \frac{0.1701z - 0.1673}{z - 1}, \quad P(z) = \frac{0.3379z + 0.2793}{z^2 - 1.5462z + 0.5646}. \quad (11)$$

Thereby, the transfer function to be identified $G(z)$ – that is also the open-loop transfer function of the NCS – is defined by (12):

$$G(z) = C(z)P(z) = \frac{g_1 z^2 + g_2 z + g_3}{z^3 + g_4 z^2 + g_5 z + g_6}, \quad (12)$$

wherein $g_1 = 0.0575$, $g_2 = -0.0090$, $g_3 = -0.0467$, $g_4 = -2.5462$, $g_5 = 2.1108$ and $g_6 = -0.5646$. The sample rate of the system is 50 samples/s and the set point $r(k)$ is an unitary step function. Network delay and packet loss are not taken into account in the simulations of this paper.

The structure of the equations (11), and so the structure of (12), are previously known by the attacker once that, as a premise, it is known that the target is an NCS that controls a DC motor using a PI controller. Thus, in these simulations, the goal of the Active System Identification attack is to discover g_1, g_2, g_3, g_4, g_5 and g_6.

The chosen attack signal $a(k)$ is a discrete-time unit impulse (13):

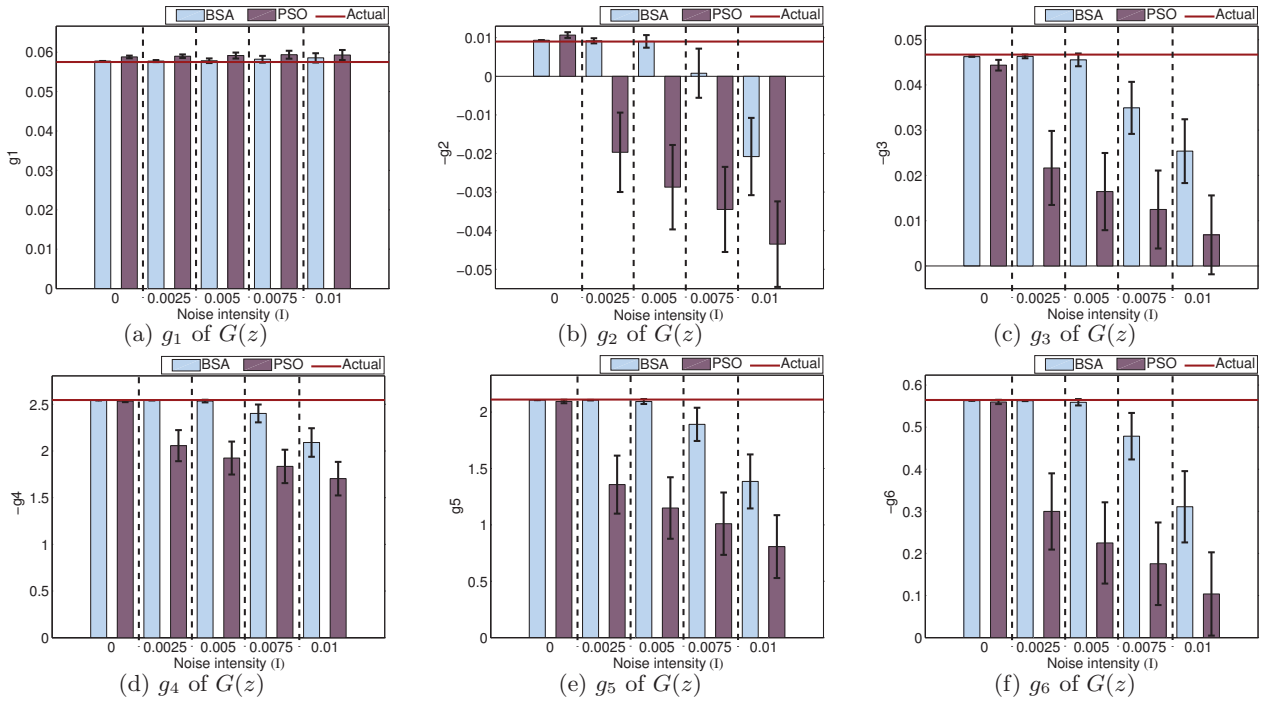
$$a(k) = \begin{cases} 1 & \text{if } k = k_a; \\ 0 & \text{otherwise,} \end{cases} \quad (13)$$

wherein k_a is the single sample in which the attacker interfere in the system by adding 1 to the feedback stream. Note that the discrete-time unit impulse is chosen to excite the NCS due to its short active time – *i.e.* one sample –, which increases the stealthiness of the attack in the time domain.

The effectiveness of the Active System Identification attacks are evaluated in both conditions with and without noise. To simulate the noise, it is inserted $w(k) \sim N(\mu, \sigma)$, indicated in Figure 2, which is a white Gaussian noise wherein N is a normal distribution, μ is its mean and σ is its standard deviation. In all simulations the mean is $\mu = 0$ rad/s. The standard deviation is adjusted such that 95% of the amplitudes of $w(k)$ are within $\pm I$ ($I = 2\sigma$). There are considered four different noise intensities I: 0 (no noise), 0.0025 rad/s, 0.005 rad/s and 0.01 rad/s. For each noise intensity I, there are executed 100 different simulations, for each of the mentioned metaheuristics. In each simulation, the feedback stream is captured by the attacker during a period $T = 2s$ (100 samples), starting at sample $k_a + 1$.

The attack model was implemented in MATLAB, where the simulations were carried out. The SIMULINK tool was used to compute $y_a(k)$ and $\hat{y}_{aj}(k)$ – the latter, for each individual j of the optimization algorithms. The parameters of the BSA and PSO described in Sections 3 and 4, respectively, were empirically adjusted through a set of simulations without noise ($I = 0$). These parameters are then used for all noise conditions. In the BSA-based attacks, the parameter η is set to 1. In the PSO-based attacks, it is used the following parameters configuration: $\omega = 0.4$, $\varphi_1 = \varphi_2 = 1.5$ and $\delta = 0.1$. In both algorithms, the population is set to 100 individuals and the limits of each dimension of the search space are $[-10, 10]$. In each simulation, the BSA and the PSO are executed for 4500 iterations.

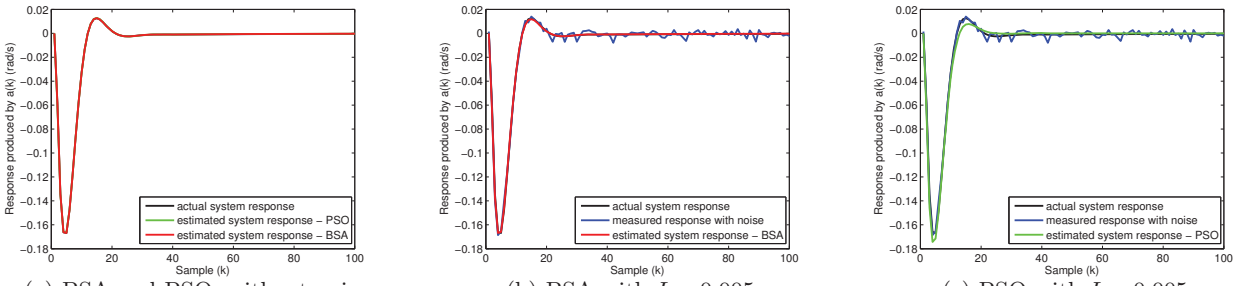
Figure 3 presents the mean estimated values of g_1, g_2, g_3, g_4, g_5 and g_6, with a Confidence Interval (CI) of 95%, for different values of noise intensity I. Note that the actual values of these coefficients are also depicted in Figure 3. In



(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) g_5 of $G(z)$ (f) g_6 of $G(z)$

Figure 3: Mean of the estimated coefficients of $G(z)$, with CI of 95%, in face of different noise intensities I.



(a) BSA and PSO, without noise (b) BSA with $I = 0.005$ (c) PSO with $I = 0.005$

Figure 4: Response of actual and estimated systems produced by $a(k)$, in face of different noise intensities.

this Figure, it is possible to compare the results achieved by the BSA-based and the PSO-based attacks. For the computation of each outcome presented in Figure 3, there were not taken into account the results beyond two standard deviation from the mean of each set of 100 simulations. According with Figure 3, it is possible to verify that, for all coefficients of $G(z)$, both the BSA-based and PSO-based attacks present good accuracy when $I = 0$ (*i.e.* without noise, the mean values of the estimated coefficients are close to their actual values). Despite the similar and accurate performance of the two metaheuristics without noise, it is possible to state that the BSA presented a performance slightly better than the PSO in this noise condition ($I = 0$), specially with regard to the coefficients g_1, g_2 and g_3. Note that, the performance of the PSO-based attack is degraded when noise is added to the system. This performance degradation of the PSO occurs for $I \geq 0.0025$, and tends to be more expressive with the increase of I. On the other hand, from Figure 3, it is possible to verify that the BSA-based attack still present good accuracy for noise intensities up to 0.005. When $I \leq 0.005$, all coefficients estimated by the BSA-based attack present a mean close to its actual value, with a small CI. When $I \geq 0.0075$, the performance of the BSA-based attack decreases with the raise of noise in a more expressive way, being worst when $I = 0.01$. Among the six coefficients of $G(z)$, in general, the estimation of g_2 presents the lowest accuracy for both BSA-based and PSO-based attacks. We attribute this behavior to a lower sensitivity that the output $\hat{y}_a(k)$ of the estimated system has to the variation of g_2. This means that, in this problem, f_j grows faster for errors in g_1, g_3, g_4, g_5 and g_6 than for errors in g_2, making the BSA population converge less accurately in dimension g_2.

The performance of the attacks can also be evaluated in the k domain through the exemples provided in Figure 4,

considering two different intensities of noise: without noise, in Figure 4(a); and with $I = 0.005$, in Figures 4(b) and 4(c). In Figure 4(a), its is shown that, without noise, the response of the system estimated by both BSA-based and PSO-based attacks matches the response of the actual system, with high accuracy. In Figure 4(b), even with a noise intensity of $I = 0.005$, the response of the system estimated by the BSA-based attack still matches the response of the actual system, indicating the convergence of $G_e(z)$ to $G(z)$ and ratifying the statistics shown in Figure 3 for the BSA with such noise intensity. On the other hand, when applying the PSO-based attack with the same noise, as exemplified in Figure 4(c), there is a slight difference between the response of the estimated system and the response of the actual system, produced by the mismatch of the estimated coefficients in the presence of such noise intensity. This exemplifies the worst performance of the PSO-based attacks when compared with the BSA-based attacks in face of the same noise intensities.

To synthesize the error of each solution found, it is computed $|E_g|$ according with (14):

$$|E_g| = \sqrt{\sum_{i=1}^{6} (g_i - g_{ei})^2}, \tag{14}$$

wherein g_i and g_{ei} are the actual and estimated coefficients of the attacked system, respectively, and i is the index number of each of the six coefficients of the model being assessed. Note that $|E_g|$ is the module of a vector composed by the error of each coefficient found, which represents another metric to evaluate the performance of each attack. The histograms of $|E_g|$ are presented in Figure 5, considering the mentioned noise intensities. It graphically shows that higher values of $|E_g|$ tend to appear more frequently as the noise intensity grows, in both BSA-based and PSO-based attacks. However, based on these histograms it is possible to verify that the mode of $|E_g|$ is close to zero for all noise intensities, using both metaheuristics. This indicates that, even in the presence of noise, most solutions present low deviations from the actual coefficients. Note that, for all noise intensities, the BSA-based attacks provide more results in the modal class – where $|E_g|$ is close to zero – than the PSO-based attacks. Moreover, the worst results of the BSA-based attacks have an $|E_g|$ about 4, when $I \geq 0.005$, while the worst results of the PSO-based attacks have an $|E_g| > 20$, when $I \geq 0.0025$. These results, together with the statistics shown in Figure 3, indicate that the performance of the Active System Identification attack is better when implemented with the BSA than with the PSO. It is worth mentioning that, to achieve these results, the BSA-based attacks consumed an average processing time $(6.68 \pm 0.47)\%$ higher than the PSO-based attacks.

In general, the outcomes indicate that, for the same amplitude of attack signal $a(k)$, the performance of the attack tends do decrease as the noise intensity increases, *i.e.* when the attack signal-to-noise ratio decreases. The minimum length of the attack signal in terms of number of manipulated samples, *i.e.* one single sample, improves the stealthiness of the attack in the k domain. On the other hand, a minimum attack signal-to-noise ratio required to guarantee the performance of this attack is a drawback with respect

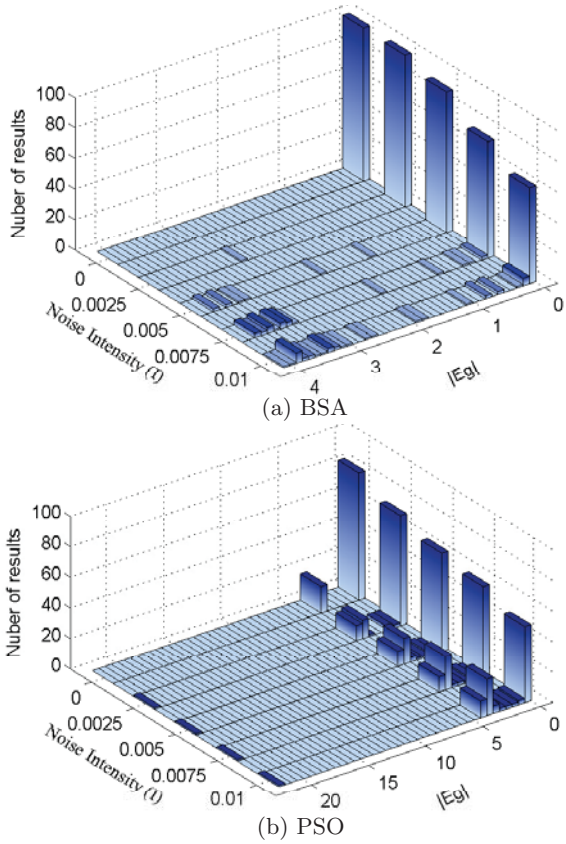


Figure 5: Histograms of $|E_g|$ for different noise intensities.

to its stealthiness, from the attacker's point of view. This issue makes more difficult for the attacker to approximate the amplitude of $a(k)$ from the noise amplitude, or to noise values that have higher probability to occur, which should help to increase the stealthiness of the attack signal in terms of amplitude.

7. CONCLUSION

The present work defines and propose an Active System Identification attack that may be launched over NCSs, in order to gather the data required for the design of other sophisticated cyber-physical attacks. The attack herein proposed is implemented based on two bio-inspired algorithms: the BSA and the PSO. It is shown that, in this problem, the BSA-based attacks provide better performance than the PSO-based attacks, specially in the presence of noise.

In general, the results indicate that the attack is capable to estimate the coefficients of the open-loop transfer function of an NCS, which is known to be enough for further manipulation of the system's behavior through conventional root locus analysis/modification. It is demonstrated the capability of the attack to achieve its goal even when:

- no meaningful information is passing through its communication links, *i.e.* when the system had achieved its steady state;
- the attacker intercepts the communication of the NCS at only one point, *i.e.* the attacker does not need to in-

tercept both forward and feedback streams to estimate the open-loop transfer function of the system;

- the NCS is noisy (particularly the BSA-based attack, for $0 \leq I \leq 0.0075$).

For future work we plan to investigate possible techniques that guarantee the performance of the attack even with small attack signal-to-noise ratio. Also, we plan – and encourage other researches – to investigate countermeasures to identify and prevent Active System Identification attacks.

8. ACKNOWLEDGMENT

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

9. REFERENCES

[1] S. Amin, X. Litrico, S. Sastry, and A. M. Bayen. Cyber security of water scada systems part i: analysis and experimentation of stealthy deception attacks. *IEEE Transactions on Control Systems Technology*, 21(5):1963–1970, 2013.

[2] E. Bou-Harb, M. Debbabi, and C. Assi. Cyber scanning: a comprehensive survey. *IEEE Communications Surveys & Tutorials*, 16(3):1496–1519, 2014.

[3] X. Chen, Y. Song, and J. Yu. Network-in-the-loop simulation platform for control system. In *AsiaSim 2012*, pages 54–62. Springer, 2012.

[4] P. Civicioglu. Backtracking search optimization algorithm for numerical optimization problems. *Applied Mathematics and Computation*, 219(15):8121–8144, 2013.

[5] A. O. de Sá, L. F. R. d. C. Carmo, and R. C. S. Machado. Covert attacks in cyber-physical control systems. *to appear in IEEE Transactions on Industrial Informatics, available at https://arxiv.org/abs/1609.09537*, arXiv:1609.09537, 2016.

[6] M. El-Sharkawi and C. Huang. Variable structure tracking of dc motor for high performance applications. *Energy Conversion, IEEE Transactions on*, 4(4):643–650, 1989.

[7] A. A. Farooqui, S. S. H. Zaidi, A. Y. Memon, and S. Qazi. Cyber security backdrop: A scada testbed. In *Computing, Communications and IT Applications Conference (ComComAp), 2014 IEEE*, pages 98–103. IEEE, 2014.

[8] N. V. George and G. Panda. A particle-swarm-optimization-based decentralized nonlinear active noise control system. *IEEE Transactions on Instrumentation and Measurement*, 61(12):3378–3386, 2012.

[9] D. Guha, P. K. Roy, and S. Banerjee. Application of backtracking search algorithm in load frequency control of multi-area interconnected power system. *Ain Shams Engineering Journal*, 2016.

[10] R. Kennedy, J. e Eberhart. Particle swarm optimization. In *Proceedings of 1995 IEEE International Conference on Neural Networks*, pages 1942–1948, 1995.

[11] R. Langner. Stuxnet: Dissecting a cyberwarfare weapon. *Security & Privacy, IEEE*, 9(3):49–51, 2011.

[12] M. Long, C.-H. Wu, and J. Y. Hung. Denial of service attacks on network-based control systems: impact and mitigation. *Industrial Informatics, IEEE Transactions on*, 1(2):85–96, 2005.

[13] R.-E. Precup, A.-D. Balint, M.-B. Radac, and E. M. Petriu. Backtracking search optimization algorithm-based approach to pid controller tuning for torque motor systems. In *Systems Conference (SysCon), 2015 9th Annual IEEE International*, pages 127–132. IEEE, 2015.

[14] Y. Shi, J. Huang, and B. Yu. Robust tracking control of networked control systems: application to a networked dc motor. *IEEE Transactions on Industrial Electronics*, 60(12):5864–5874, 2013.

[15] M. L. Si, H. X. Li, X. F. Chen, and G. H. Wang. Study on sample rate and performance of a networked control system by simulation. In *Advanced Materials Research*, volume 139, pages 2225–2228. Trans Tech Publ, 2010.

[16] R. Smith. A decoupled feedback structure for covertly appropriating networked control systems. In *Proceedings of the 18th IFAC World Congress 2011*, volume 18. IFAC-PapersOnLine, 2011.

[17] R. S. Smith. Covert misappropriation of networked control systems: Presenting a feedback structure. *Control Systems, IEEE*, 35(1):82–92, 2015.

[18] A. C. Snoeren, C. Partridge, L. A. Sanchez, C. E. Jones, F. Tchakountio, B. Schwartz, S. T. Kent, and W. T. Strayer. Single-packet ip traceback. *IEEE/ACM Transactions on Networking (ToN)*, 10(6):721–734, 2002.

[19] W. Stallings. *Cryptography and network security: principles and practices*. Pearson Education India, 2006.

[20] A. Teixeira, I. Shames, H. Sandberg, and K. H. Johansson. A secure control framework for resource-limited adversaries. *Automatica*, 51:135–148, 2015.

[21] T. Tran, Q. P. Ha, and H. T. Nguyen. Robust non-overshoot time responses using cascade sliding mode-pid control. *Journal of Advanced Computational Intelligence and Intelligent Informatics*, 2007.

[22] H. J. Tulleken. Generalized binary noise test-signal concept for improved identification-experiment design. *Automatica*, 26(1):37–49, 1990.

[23] S. Uong and I. Ngamroo. Coordinated control of dfig wind turbine and svc for robust power system stabilization. In *Electrical Engineering/Electronics, Computer, Telecommunications and Information Technology (ECTI-CON), 2015 12th International Conference on*, pages 1–6. IEEE, 2015.

[24] W. Xin, L. Ran, W. Yanghua, P. Yong, and Q. Bin. Self-tuning pid controller with variable parameters based on particle swarm optimization. In *Intelligent System Design and Engineering Applications (ISDEA), 2013 Third International Conference on*, pages 1264–1267. IEEE, 2013.