Kỷ yếu Hội nghị Khoa học Quốc gia lần thứ IX “Nghiên cứu cơ bản và ứng dụng Công nghệ thông tin (FAIR'9)”; Cần Thơ, ngày 4-5/8/2016
DOI: 10.15625/vap.2016.0002

A NEW METHOD AGAINST ATTACKS ON NETWORKED INDUSTRIAL CONTROL SYSTEMS

Nguyen Dao Truong, Le My Tu
Academy of Cryptography Technique
truongnguyendao@gmail.com, tulemy@hotmail.com

ABSTRACT *— In this paper, by incorporating knowledge of the physical system under control, we proposed the new method to detect computer attacks that change the behavior of the targeted industrial control system. By using knowledge of the physical system we are able to focus on the final objective of the attack, and not on the particular mechanisms of how vulnerabilities are exploited, and how the attack is hidden. We also analyze the safety of our solution by exploring the effects of stealthy attacks, and by hopping that automatic attack-response mechanisms will not drive the system to an unsafe state. In our paper, we proposed two module: one for changing detection by sequential detection and CUSUM statistic, other one for responsing attacks by linear model predictive control algorithm to keep the system in safety state before human operators can control the system.*

Keywords *— Linear model predictive control, control system, model algorithmic control, dynamic matrix control, attack.*

I. INTRODUCTION

Networked industrial control systems are computer-based systems that monitor and control physical processes. These systems represent a wide variety of networked information technology (IT) systems connected to the physical world. Depending on the application, these control systems are also called Process Control Systems (PCS), Supervisory Control and Data Acquisition (SCADA) systems, Distributed Control Systems (DCS) or Cyber-Physical Systems (CPS). Control systems are usually composed of a set of networked agents, consisting of sensors, actuators, control processing units such as programmable logic controllers (PLCs), and communication devices. Modern day industrial control systems have a multi-layer structure [11]. The overall objectives of such a control structure are: (1) to maintain safe operational goals by limiting the probability of undesirable behavior, (2) to meet the production demands by keeping certain process values within prescribed limits, (3) to maximize production profit. Several control applications can be labeled as safety-critical: their failure can cause irreparable harm to the physical system being controlled and to the people who depend on it. SCADA systems, in particular, perform vital functions in national critical infrastructure systems, such as electric power distribution, oil and natural gas distribution, water and waste-water treatment, and transportation systems. The disruption of these control systems could have a significant impact on public health, safety and lead to large economic losses. Control systems have been at the core of critical infrastructures, manufacturing and industrial plants for many decades, and yet, there have been few confirmed cases of cyber attacks. Control systems, however, are now at a higher risk to computer attacks because their vulnerabilities are increasingly becoming exposed and available to an ever-growing set of motivated and highly-skilled attackers.

An accurate assessment of potential losses under cyber-attacks is a pre-requisite for any risk management program. Risk management is the process of shifting the odds in your favor by finding among all possible alternatives, the one that minimizes the impact of uncertain events. Probably the best well known risk metric is the average loss $R_\mu = \mathbb{E}[\mathrm{L}] \approx \sum_i L_i p_i$, where L_i is the loss if event i occurs, and p_i is the probability that event i occurs. Other risk metrics try to get more information about the probability distribution of the losses, and not only its mean value (R_μ). For example the variance of the losses $R_\chi = \mathbb{E}[\mathrm{L}^2]\text{-}\mathrm{R}_\mu$ is very useful in finance since it gives more information to risk averse individuals. This is particularly important if the average loss is computed for a large period of time (e.g. annually). If the loss is considered every time there is a computer event then we believe the average loss by itself provides enough risk information to make a rational decision.

In this paper, we focus on attacks on sensor networks and the effects they have on the process control system. Therefore p_i denotes the likelihood that an attacker will compromise sensor i, and L_i denotes the losses associated with that particular compromise. To simplify our presentation we assume that p_i is the same for all sensors, therefore our focus in the remaining of this paper is to estimate the potential losses L_i. The results can then be used to identify high priority sensors and to invest a given security budget in the most cost effective way.

II. DETECTION OF ATTACKS ON NETWORED INDUSTRIAL CONTROL SYSTEMS

A. Attack models

We consider the case when the state of the system is measured by a sensor network of l sensors with measurement vector $y(k) = \{y_1(k), y_2(k), ..., y_l(k)\}$, where $y_i(k)$ denotes the measurement by sensor i at time k. All sensors have a dynamic range that defines the domain of y_i for all k. That is, all sensors have defined minimum and maximum values $\forall k, y_i(k) \in \left[y_i^{\min}, y_i^{\max}\right]$. Let $Y_i = \left[y_i^{\min}, y_i^{\max}\right]$. We assume each sensor has a unique identity protected by a cryptographic key. Let $\tilde{y}(k) \in \mathbb{R}^l$ denote the received measurements by the controller at time k. Based on these measurements the control system defines control actions to maintain certain operational goals. If some of the sensors are under attack, $\tilde{y}(k)$ may be different from the real measurement $y(k)$; however, we assume that the attacked signals $\tilde{y}_i(k)$ also lie within Y_i (signals outside this range can be easily detected by fault-tolerant algorithms).

Let $K_a = \{k_s, ..., k_e\}$ represent the attack duration; between the start time k_s and stop time k_e of an attack. A general model for the observed signal is the following:

$$y_i(k) = \begin{cases} y_i(k) & fork \notin K_a \\ a_i(k) & fork \in K_a, a_i(k) \in Y_i \end{cases} \tag{1}$$

where $a_i(k)$ is the attack signal. This general sensor attack model can be used to represent integrity attacks and DoS attacks. In an integrity attack we assume that if attackers have compromised a sensor, then they can inject any arbitrary value, therefore in this case, $a_i(k)$ is some arbitrary non-zero value. In a DoS attack, the controller will notice the lack of new measurements and will react accordingly. An intuitive response for a controller to implement against a DoS attack is to use the last signal received: $a_i(k) = y_i(k_s)$, where $y_i(k_s)$ is the last measurement received before the DoS attack starts.

B. Linear model predictive control

Linear Model predictive control (LMPC) algorithms employ linear or linearized models to obtain the predictive response of the controlled process. These algorithms include the Model Algorithmic Control (MAC)[12], the Dynamic Matrix Control (DMC)[3] and the Generalized Predictive Control (GPC)[2]. These algorithms are all similar in the sense that they rely on process models to predict the behavior of the process over some future time interval, and the control calculations are based on these model predictions. The models used for these predictions have usually been derived from linear approximations of the process or experimentally obtained step response data. A survey of theory and applications of such algorithms have been reported in [6].

A classical autoregressive moving average (ARX) model structure that relates the plant output with the present and past plant input-output can be used to formulate a predictive model. The model parameters can be determined a priori by using the known input-output data to form a fixed predictive model or these parameters are updated at each sampling time by an adaptive mechanism. The one step ahead predictive model can be recursively extended to obtain future predictions for the plant output. The minimization of a cost function based on future plant predictions and desired plant outputs generates an optimal control input sequence to act on the plant. The strategy is described as follows.

1. Predictive model

The relation between the past input-output data and the predicted output can be expressed by an ARX model of the form

$$y(t+1) = a_1 y(t) + ... + a_{ny} y(t-ny+1) + b_1 u(t) + ... + b_{nu} u(t-nu+1) \tag{2}$$

where $y(t)$ and $u(t)$ are the process and controller outputs at time t, $y(t+1)$ is the one-step ahead model prediction at time t, a's and b's represent the model coefficients and the nu and ny are input and output orders of the system.

2. Model identification

The model output prediction can be expressed as $y_m(t+1) \neq \phi x_m(t)$ (3)

Where $\phi = [\alpha_1 ... \alpha_{ny} \beta_1 ... \beta_{nu}]$

and
$$x_m(t+1) = [y(t)...y(t-ny+1)u(t)...u(t-nu+1)]^T \tag{4}$$

with α and β as identified model parameters.

One of the most widely used estimators for model parameters and covariance is the popular recursive least squares (RLS) algorithm[7]. The RLS algorithm provides the updated parameters of the ARX model in the operating space at each sampling instant or these parameters can be determined a priority using the known data of inputs and outputs for different operating conditions. The RLS algorithm is expressed as

$$\begin{aligned} &\phi(t+1) = \phi(t) + K(t)[y(t+1) - y_m(t+1)] \\ &\mathrm{K(t)=P(t)x_m(t+1)/[1+ x_m(t+1)}^T P(t) \mathrm{x_m(t+1)]} \\ &P(t+1) = 1/\lambda[P(t) - \{(P(t)\mathrm{x_m(t+1)x_m(t+1)}^T P(t)) / (1 + \mathrm{x_m(t+1)}^T P(t) \mathrm{x_m(t+1)})\}] \end{aligned} \tag{5}$$

where $\phi(t)$ represents the estimated parameter vector, λ is the forgetting factor, $K(t)$ is the gain matrix and $P(t)$ is the covariance matrix.

3. Controller formulation

The *N* time steps ahead output prediction over a prediction horizon is given by

$$y_l(t+N) = a_1 y(t+N-1) + ... + a_{ny} y(t-ny+N) + \beta_1 u(t+N-1) + ... + \beta_{nu} u(t-nu+N) + err(t) \quad (6)$$

where $y_l(t+N)$ represent the model predictions for *N* steps and $err(t)$ is an estimate of the modeling error which is assumed as constant for the entire prediction horizon. If the control horizon is *m*, then the controller output, *u* after *m* time steps can be assumed to be constant.

An internal model is used to eliminate the discrepancy between model and process outputs, $error(t)$, at each sampling instant

$$error(t) = y(t) - y_m(t) \quad (7)$$

where $y_m(t)$ is the one-step ahead model prediction at time (*t*-1). The estimate of the error is then filtered to produce $err(t)$ which minimizes the instability introduced by the modeling error feedback. The filter error is given by

$$err(t) = (1-K_f)err(t-1) + K_f err(t) \quad (8)$$

where K_f is the feedback filter gain which has to be tuned heuristically.

Back substitutions transform the prediction model equations into the following form

$$y_p(t+N) = f_{N,1} y(t) + ... + f_{N,ny} y(t-ny+1) + f_{N,ny+1} u(t-1) + ... + f_{N,ny+nu-1} u(t-nu+1) + g_{N,1} u(t) + ... + g_{N,m} u(t+m-1) + e_N err(t) \quad (9)$$

The elements *f*, *g* and *e* are recursively calculated using the parameters α and β of (4). The above equations can be written in a condensed form as

$$Y(t) = FX(t) + GU(t) + Eerr(t) \quad (10)$$

Where $Y(t) = [y_l(t+1)...y_l(t+N)]^T$, $X(t) = [y(t)y(t-1)...y(t-ny+1)u(t-1)...u(t-nu+1)]^T$, $U(t) = [u(t)...u(t+m-1)]^T$

$$F = \begin{bmatrix} f_{11} & f_{12} & \cdots & f_{1(ny+nu-1)} \\ f_{21} & f_{22} & \cdots & f_{2(ny+nu-1)} \\ & & \cdots & \\ & & \cdots & \\ f_{N1} & f_{N2} & \cdots & f_{N(ny+nu-1)} \end{bmatrix} \quad G = \begin{bmatrix} g_{11} & 0 & 0 & ... & 0 \\ g_{21} & g_{22} & 0 & ... & 0 \\ ... & ... & ... & ... & ... \\ g_{m1} & g_{m2} & g_{m3} & \cdots & g_{mm} \\ ... & ... & ... & ... & ... \\ g_{N1} & g_{N2} & g_{N3} & \cdots & g_{Nm} \end{bmatrix} \quad E = [e_1 ... e_N]^T$$

In the above, $Y(t)$ represents the model predictions over the prediction horizon, $X(t)$ is a vector of past plant and controller outputs and $U(t)$ is a vector of future controller outputs. If the coefficients of *F*, *G* and *E* are determined then the transformation can be completed. The number of columns in *F* is determined by the ARX model structure used to represent the system, where as the number of columns in *G* is determined by the length of the control horizon. The number of rows is fixed by the length of the prediction horizon.

Consider a cost function of the form

$$J = \sum_{i=1}^{N} [y_l(t+i) - w(t+i)]^2 + \sum_{i=1}^{m} \gamma [u(t+i-1)]^2 = \sum_{i=1}^{N} [Y(t) - W(t)]^T [Y(t) - W(t)] + \sum_{i=1}^{m} \gamma U(t)^T U(t) \quad (11)$$

where $W(t)$ is a setpoint vector over the prediction horizon

$$W(t) = [w(t+1)...w(t+N)]^T \quad (12)$$

The minimization of the cost function, *J* gives optimal controller output sequence

$$U(t) = [G^T G + \gamma I]^{-1} G^T [W(t) - FX(t) - Eerr(t)] \quad (13)$$

The vector $U(t)$ generates control sequence over the entire control horizon. But, the first component of $U(t)$ is actually implemented and the whole procedure is repeated again at the next sampling instant using latest measured information.

Linear model predictive control involving input-output models in classical, adaptive or fuzzy forms is proved useful for controlling processes that exhibit even some degree of nonlinear behavior [5], [15], [16].

In this paper, we used combined system model TE-PCS (Tennessee-Eastman Process Control System) with loop control laws PI which proposed by Ricker[13]. General processing and control processes described in Fig 1.

C. Detection of attacks

Detecting attacks to control systems can be formulated as an anomaly-based intrusion detection problem [4]. One big difference in control systems compared to traditional IT systems, is that instead of creating models of network traffic or software behavior, we can use a representative model of the physical system. The intuition behind this approach is the following: if we know how the output sequence of the physical system, $y(k)$, should react to the control input sequence, $u(k)$, then any attack to the sensor data can be potentially detected by comparing the expected output $\hat{y}(k)$ with the received (and possibly compromised) signal $\tilde{y}(k)$. Depending on the quality of our estimate $\hat{y}(k)$ we may have some false alarms.

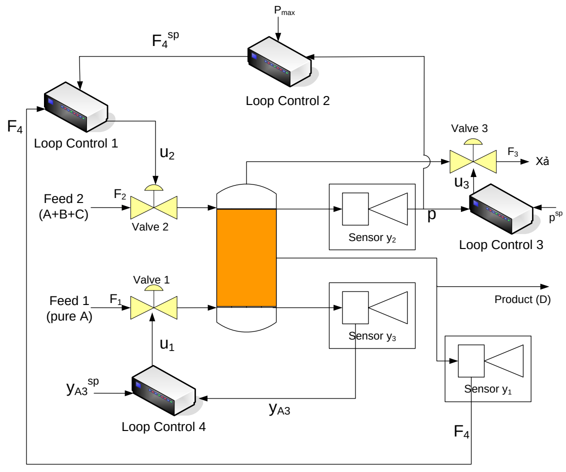

Fig 1. Typical Industrial Control System

To formalize the anomaly detection problem, we need (1) a model of the behavior of the physical system, and (2) an anomaly detection algorithm. In section II.B we discuss our choice of linear model predictive control as an approximation of the behavior of the physical system. In this section, we describe change detection theory and the detection algorithm we use a nonparametric cumulative sum (CUSUM) statistic.

The physical-model-based attack detection method presented in this paper can be viewed as complementary to intrusion detection methods based on network and computer systems models. Because we need to detect anomalies in real time, we can use results from sequential detection theory to give a sound foundation to our approach. Sequential detection theory considers the problem where the measurement time is not fixed, but can be chosen online as and when the measurements are obtained. Such problem formulations are called optimal stopping problems. Two such problem formulations are: sequential detection (also known as sequential hypothesis testing), and quickest detection (also known as change detection). A good survey of these problems is given by Kailath and Poor [10].

In optimal stopping problems, we are given a time series sequence $z(1)$, $z(2)$,..., $z(N)$, and the goal is to determine the minimum number of samples, N, the anomaly detection scheme should observe before making a decision d_N between two hypotheses: H_0 (normal behavior) and H_1 (attack). The difference between sequential detection and change detection is that the former assumes the sequence $z(i)$ is generated either by the normal hypothesis (H_0), or by the attack hypothesis (H_1). The goal is to decide which hypothesis is true in minimum time. On the other hand, change detection assumes the observation $z(i)$ starts under H_0 and then, at a given ks it changes to hypothesis H_1. Here the goal is to detect this change as soon as possible. Both problem formulations are very popular, but security researchers have used sequential detection more frequently. However, for our attack detection method, the change detection formulation is more intuitive. To facilitate this intuition, we now briefly describe the two formulations.

1. Sequential Detection

Given a fixed probability of false alarm and a fixed probability of detection, the goal of sequential detection is to minimize the number of observations required to make a decision between two hypotheses. The solution is the classic sequential probability ratio test (SPRT) of Wald[17] (also referred as the threshold random walk (TRW) by some security papers). SPRT has been widely used in various problems in information security such as detecting portscans[9], worms[14], proxies used by spammers[18], and botnets[8].

Assuming that the observations $z(k)$ under H_j are generated with a probability distribution p_j, the SPRT algorithm can be described by the following equations:

$$\begin{aligned} S(k+1) &= \log \frac{p_1(z(k))}{p_0(z(k))} + S(k) \\ N &= \inf_n \{\text{n:S(n)} \notin [L, U]\} \end{aligned} \tag{14}$$

starting with $S(0) = 0$. The SPRT decision rule d_N is defined as: $d_N = \begin{cases} H_1 \text{ if } S(N) \geq U \\ H_0 \text{ if } S(N) \leq L \end{cases}$ (15)

where $L \approx \ln \frac{b}{1-a}$ and $U \approx \ln \frac{1-b}{a}$ and where a is the desired probability of false alarm and *b* is the desired probability of missed detection (usually chosen as small values).

2. Change Detection

The goal of the change detection problem is to detect a possible change, at an unknown change point k_s. Cumulative sum (CUSUM) and Shiryaev-Roberts statistics are the two most commonly used algorithms for change detection problems. In this paper, we use the CUSUM statistic because it is very similar to the SPRT.

Given a fixed false alarm rate, the CUSUM algorithm attempts to minimize the time *N* (where $N > k_s$) for which the test stops and decides that a change has occurred. Let $S(0) = 0$. The CUSUM statistic is updated according to

$$S(k+1) = \left(\log \frac{p_1(z(k))}{p_0(z(k))} + S(k) \right)^+ \qquad (16)$$

where $(a)^+ = a$ if $a \geq 0$ and zero otherwise. The stopping time is: $N = \inf_n \{n : S(n) \geq \tau\}$ (17)

for a given threshold τ selected based on the false alarm constraint.

We can see that the CUSUM algorithm is an SPRT test with $L = 0$, $U = \tau$, and whenever the statistic reaches the lower threshold *L*, it restarts. We now describe how to adapt the results of change detection theory to the particular problem of detecting compromised sensors. In the following, we use the subscript *i* to denote the sequence corresponding to sensor *i*.

One problem that we have in our case is that we do not know the probability distribution for an attack p_1. In general, an adaptive adversary can select any arbitrary (and possibly) non-stationary sequence $z_i(k)$. Assuming a fixed p_1 will thus limit our ability to detect a wide range of attacks.

To avoid making assumptions about the probability distribution of an attacker, we use ideas from nonparametric statistics. We do not assume a parametric distribution for p_1 and p_0; instead, only place mild constraints on the observation sequence. One of the simplest constraints is to assume the expected value of the random process $Z_i(k)$ that generates the sequence $z_i(k)$ under H_0 is less than zero ($\mathbb{E}_0[Z_i] < 0$) and the expected value of $Z_i(k)$ under H_1 is greater than zero ($\mathbb{E}_1[Z_i] > 0$).

To achieve these conditions let us define $z_i(k) = \|\tilde{y}_i(k) - \hat{y}_i(k)\| - b_i$ (18)

where b_i is a small positive constant chosen such that $\mathbb{E}_0\left[\|\tilde{y}_i(k) - \hat{y}_i(k)\| - b_i\right] < 0$ (19)

The nonparametric CUSUM statistic for sensor i is then: $S_i(k) = (S_i(k-1) + z_i(k))^+, S_i(0) = 0$ (20)

and the corresponding decision rule is $d_{N,i} = d_\tau(S_i(k)) = \begin{cases} H_1 & \text{if } S_i(k) > \tau_i \\ H_0 & otherwise \end{cases}$ (21)

where τ_i is the threshold selected based on the false alarm rate for sensor *i*.

Following [1], we state the following two important results for Equation (20)-(21):

- The probability of false alarm decreases exponentially as the threshold τ_i increases,
- The time to detect an attack, $(N_i - k_{s,i})^+$, is inversely proportional to b_i.

D. Stealthy Attacks

A fundamental problem in intrusion detection is the existence of adaptive adversaries that will attempt to evade the detection scheme; therefore, we now consider an adversary that knows about our anomaly detection scheme. We take a conservative approach in our models by assuming a very powerful attacker with knowledge of: (1) the exact linear model that we use (i.e., matrices A, B, and C), the parameters (τ_i and b_i), and (3) the control command signals. Such a powerful attacker may be unrealistic in some scenarios, but we want to test the resiliency of our system to such an attacker to guarantee safety for a wide range of attack scenarios.

The goal of the attacker is to raise the pressure in the tank without being detected (i.e., raise the pressure while keeping the statistic he controls below the corresponding threshold τ_i). We model three types of attacks: surge attacks, bias attacks and geometric attacks. Surge attacks model attackers that want to achieve maximum damage as soon as they get access to the system. A bias attack models attackers that try to modify the system discretely by adding small perturbations over a large period of time. Finally, geometric attacks model attackers that try to shift the behavior of the system very discretely at the beginning of the attack and then maximize the damage after the system has been moved to a more vulnerable state.

E. Surge Attacks

In a surge attack the adversary tries to maximize the damage as soon as possible, but when the statistic reaches the threshold, it then stays at the threshold level: $S_i(k) = \tau$ for the remaining time of the attack. To stay at the threshold, the attacker needs to solve the following quadratic equation:

$$S_i(k) + \sqrt{\left(\hat{y}_i(k) - \tilde{y}_i(k)\right)^2} - b_i = \tau_i \tag{22}$$

The resulting attack (for y_2 and y_1) is: $$\tilde{y}_i(k) = \begin{cases} y_i^{\min} & \text{if } S_i(k+1) \le \tau_i \\ \hat{y}_i(k) - \left|\tau_i + b_i + S_i(k)\right| & \text{if } S_i(k+1) > \tau_i \end{cases} \tag{23}$$

For y_3 we use $$\tilde{y}_3(k) = \begin{cases} y_3^{\min} & \text{if } S_{y_3}(k) \le \tau_3 \\ \hat{y}_3(k) - \left|\tau_3 + b_3 + S_{y_3}(k)\right| & \text{if } S_{y_3}(k) > \tau_3 \end{cases} \tag{24}$$

F. Bias Attacks

In a bias attack the attacker adds a small constant c_i at each time step.

$$\tilde{y}_{i,k} = \hat{y}_{i,k} - c_i \in Y_i \tag{25}$$

In this case, the nonparametric CUSUM statistic can be written as $$S_i(n) = \sum_{k=0}^{n-1} \left|\hat{y}_i(k) - \tilde{y}_i(k)\right| - nb_i \tag{26}$$

Assuming the attack starts at time $k = 0$ and assuming the attacker wants to be undetected for n time steps the attacker needs to solve the following equation $$\sum_{k=0}^{n-1} c_i = \tau_i + nb_i \tag{27}$$

Therefore $c_i = \tau_i/n + b$. This attack creates a bias of $\tau_i/n + b_i$ for each attacked signal.

This equation shows the limitations of the attacker. If an attacker wants to maximize the damage (maximize the bias of a signal), the attacker needs to select the smallest n it can find. Because $\tilde{y}_i \in Y_i$ this attack reduces to an impulse attack.

If an attacker wants to attack for a long time, then n will be very large. If n is very large then the bias will be smaller.

G. Geometric Attacks

In a geometric attack, the attacker wants to drift the value very slowly at the beginning and maximize the damage at the end. This attack combines the slow initial drift of the bias attack with a surge attack at the end to cause maximum damage.

Let $\alpha \in (0, 1)$. The attack is $$\tilde{y}_i(k) = \hat{y}_i(k) - \beta_i \alpha_i^{n-k} \tag{28}$$

Now we need to find α and β such that $S_i(n) = \tau_i$.

Assume the attack starts at time $k = 0$ and the attacker wants to be undetected for n time steps. The attacker then needs to solve the following equation: $$\sum_{k=0}^{n-1} \beta_i \alpha_i^{n-k} - nb_i = \tau_i \tag{29}$$

This addition is a geometric progression $$\sum_{k=0}^{n-1} \beta_i \alpha_i^{n-k} = \beta_i \alpha_i^n \sum_{k=0}^{n-1} (\alpha_i^{-1})^k = \beta_i \frac{1-\alpha_i^n}{\alpha_i^{-1} - 1} \tag{30}$$

By fixing α the attacker can select the appropriate β to satisfy the above equation.

III. RESPONSE TO ATTACKS

A comprehensive security posture for any system should include mechanisms for prevention, detection, and response to attacks. Automatic response to computer attacks is one of the fundamental problems in information assurance. While most of the research efforts found in the literature focus on prevention (authentication, access controls, cryptography etc.) or detection (intrusion detection systems), in practice there are quite a few response mechanisms.

Given that we already have an estimate for the state of the system (given by a linear model predictive control module), a natural response strategy for control systems is to use this estimate when the anomaly detection statistic fires an alarm. Fig 2 shows our proposed architecture. Specifically: for sensor *i*, if $S_i(k) > \tau_i$, the CDM (Change Detection Module) replaces the sensor measurements $\tilde{y}_i(k)$ with measurements generated by the linear model predictive control module $\hat{y}_i(k)$ (that is the controller will receive as input $\hat{y}_i(k)$ instead of $\tilde{y}_i(k)$). Otherwise, it treats $\tilde{y}_i(k)$ as the correct sensor signal.

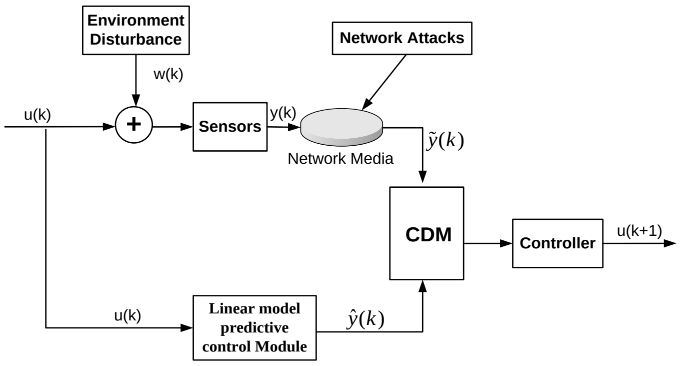

Fig 2. The proposed attack detection model

In our proposed detection and response architecture which described in Fig. 2, we hope to make sure that if there is a false alarm, controlling the system by using the estimated values from the linear model predictive control system will not cause any safety concerns. The automatic response mechanism works well when we are under attack (with $\tilde{y}_2 = 0.5 * y_2$). Fig.3 and Fig.4 shows that when an attack is detected, the response algorithm manages to keep the system in a safe state. Similar results were obtained for all detectable attacks.

While our attack response mechanism is a good solution when the alarms are indeed an indication of attacks, Our main concern in this section is the cost of false alarms. To address these concerns we ran the simulation scenario without any attacks 500 times; each time the experiment ran for 20 hours. As expected, with the parameter set $\tau_{y_1} = 5$, $\tau_{y_2} = 1000$, $\tau_{y_3} = 20$ our system did not detect any false alarm; therefore we decided to reduce the detection threshold to $\tau_{y_1} = 1$, $\tau_{y_2} = 100$, $\tau_{y_3} = 2$ and run the same experiments again. We can see that while a false response mechanism increases the pressure of the tank, it never reaches unsafe levels. The maximum pressure obtained while controlling the system based on the linear model was 278 kPa, which is in the same order of magnitude than the normal variation of the pressure without any false alarm (276 kPa).

In our case, even if the system is kept in a safe state by the automated response, our response strategy is meant as a temporary solution before a human operator responds to the alarm. Based on our results we believe that the time for a human response can be very large.

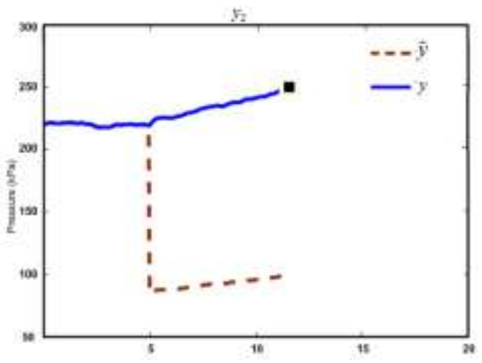

Fig 3. Without Change Detection Module (CDM)

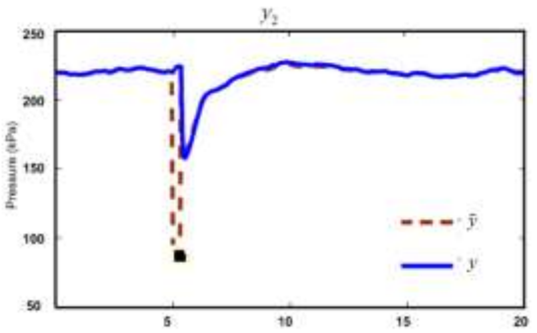

Fig 4. CDM detects and response to the attack at t=5.1

IV. CONCLUSION

In this paper we identified some research challenges for securing control systems. We showed that by incorporating a physical model of the system we were able to identify the most critical sensors and attacks. We proposed the use of automatic response mechanisms based on linear model predictive control of the system. Automatic responses may be problematic in some cases (especially if the response to a false alarm is costly); therefore, we would like to emphasize that the automatic response mechanism should be considered as a temporary solution before a human investigates the alarm. A full deployment of any automatic response mechanism should take into consideration the amount of time in which it is reasonable for a human operator to respond, and the potential side effects of responding to a false alarm.

However, as a word of caution, large scale control system designs are often not to resilient by design and may become prey to such stealth attacks if sufficient resilience is not built by design in the first place. Thus, our ideas become all the more relevant for operational security until there is a principled way of designing fully attack resilient control structures and algorithms (which by itself is a very challenging research endeavor and may not offer a cost effective design solution). By our proposed solution, we hope keep the system in safety state before the human operators can control the system and and no catastrophes occur.

V. REFERENCES

[1] Brodsky B. and Darkhovsky B., "Non-Parametric Methods in Change-Point Problems", Kluwer Academic Publishers, 1993.

[2] Clarke D. W., Mohtadi C and Tuffs P. S., "Generalized predictive control – Part I. The basic algorithm", Automatica, 23: 137-148, 1987.

[3] Cutler C. R. and Ramker B. L. "Dynamic matrix control – a computer control algorithm", Proceedings Joint Automatic Control Conference, Sanfrancisco, CA.,1980.

[4] Denning D., "An intrusion-detection model", Software Engineering, IEEE Transac- tions on SE-13(2), 1987, pp.222–232.

[5] Eaton J. W., Rawlings J. B., "Model predictive control of chemical processes", Chemical Engineering Science, 47: 705-720, 1992.

[6] Garcia C. E., Prett D. M., and Morari M., "Model predictive control: Theory and Practice - A survey", Automatica, 25: 335-348, 1989.

[7] Goodwin G. C., Sin K. S., "Adaptive Filtering Prediction and Control", Printice Hall, Englewood Cliffs, New Jersey, 1984.

[8] Gu G., Zhang J. and Lee W., "Botsniffer: Detecting botnet command and control channels in network traffic", in Proceedings of the 15th Annual Network and Distributed System Security Symposium (NDSS'08), San Diego, CA, 2008.

[9] Jung J., Paxson V., Berger A. and Balakrishan H., "Fast portscan detection using sequential hypothesis testing", in Proceedings of the 2004 IEEE Symposium on Security and Privacy, 2004, pp. 211–225.

[10] Kailath T. and Poor H. V., "Detection of stochastic processes", IEEE Transactions on Information Theory 44(6), 1998, pp.2230–2258.

[11] Quin S. J. and Badgwell T. A., "A survey of industrial model predictive control technology", Control Engineering Practice 11(7), 2003, pp.733-764.

[12] Rawlings J., "Tutorial overview of model predictive control", Control Systems Maga-zine, IEEE 20(3), 2000, pp.38–52.

[13] Richalet J., Rault A., Testud J. L. and Papon J., "Model predictive heuristic control: Application to industrial processes", Automatica, 14: 413-428, 1978.

[14] Ricker N., "Model predictive control of a continuous, nonlinear, two-phase reactor", Journal of Process Control 3(2), 1993, pp. 109–123.

[15] Schechter S. and Berger J. J. A., "Fast detection of scanning worm infections", in 'Proc. of the Seventh International Symposium on Recent Advances in Intrusion Detection (RAID)', 2004.

[16] Venkateswarlu Ch., Gangiah K., "Constrained generalized predictive control of unstable nonlinear processes", Transactions of Insitution of Chemical Engineers, 75: 371-376, 1997.

[17] Venkateswarlu Ch., Naidu K.V.S. "Adaptive fuzzy model predictive control of an exothermic batch chemical reactor", Chemical Engineering Communications, 186: 1-23, 2001.

[18] Wald A., "Sequential Analysis", J. Wiley & Sons, New York, NY, 1947.

MỘT PHƯƠNG PHÁP MỚI CHỐNG TẤN CÔNG TRONG MẠNG ĐIỀU KHIỂN CÔNG NGHIỆP

Nguyễn Đào Trường, Lê Mỹ Tú

TÓM TẮT— *Trong bài báo này, bằng cách kết hợp những kiến thức về điều khiển hệ thống vật lý, chúng tôi đề xuất một phương pháp mới để phát hiện các tấn công máy tính mà chúng làm thay đổi hành vi của hệ thống điều khiển công nghiệp. Bằng việc sử dụng những kiến thức về hệ thống vật lý chúng tôi tập trung vào phân tích các hình thức tấn công, những lỗ hổng bị khai thác không theo những cơ chế đặc thù mà từ đó kẻ tấn công có thể ẩn lấp. Chúng tôi phân tích sự an toàn của giải pháp đề xuất thông qua thực thi các hình thức tấn công và hy vọng giải pháp phản ứng tấn công tự động sẽ đưa hệ thống về trạng thái an toàn trước các tấn công. Trong bài báo này, chúng tôi đề xuất sử dụng hai module: một cho phát hiện những thay đổi bằng phương pháp phát hiện tuần tự và thống kê CUSUM, còn lại để phản ứng lại các tấn công bằng thuật toán điều khiển tiên đoán tuyến tính để giữ cho hệ thống ở trạng thái an toàn trước khi người vận hành có thể kiểm soát được tình hình.*