Attacks Against Process Control Systems: Risk Assessment, Detection, and Response

Alvaro A. Cárdenas[§], Saurabh Amin[‡], Zong-Syun Lin[†],
Yu-Lun Huang[†], Chi-Yen Huang[†] and Shankar Sastry[‡]

[§] Fujitsu Laboratories of America
[‡] University of California, Berkeley
[†] National Chiao Tung University, Taiwan

ABSTRACT

In the last years there has been an increasing interest in the security of process control and SCADA systems. Furthermore, recent computer attacks such as the Stuxnet worm, have shown there are parties with the motivation and resources to effectively attack control systems.

While previous work has proposed new security mechanisms for control systems, few of them have explored new and fundamentally different research problems for securing control systems when compared to securing traditional information technology (IT) systems. In particular, the sophistication of new malware attacking control systems–malware including zero-days attacks, rootkits created for control systems, and software signed by trusted certificate authorities–has shown that it is very difficult to prevent and detect these attacks based solely on IT system information.

In this paper we show how, by incorporating knowledge of the physical system under control, we are able to detect computer attacks that change the behavior of the targeted 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 analyze the security and safety of our mechanisms by exploring the effects of stealthy attacks, and by ensuring that automatic attack-response mechanisms will not drive the system to an unsafe state.

A secondary goal of this paper is to initiate the discussion between control and security practitioners–two areas that have had little interaction in the past. We believe that control engineers can leverage security engineering to design–based on a combination of their best practices–control algorithms that go beyond safety and fault tolerance, and include considerations to survive targeted attacks.

Categories and Subject Descriptors

C.2.0 [**Computer-Communication Network**]: Security and Protection; B.8.2 [**Performance and Reliability**]: Performance Analysis and Design Aids

General Terms

Security

Keywords

SCADA, security, IDS, control systems, critical infrastructure protection, cyber-physical systems

1. INTRODUCTION

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 Aquisition (SCADA) systems (in industrial control or in the control of the critical infrastructures), Distributed Control Systems (DCS) or Cyber-Physical Systems (CPS) (to refer to embedded sensor and actuator networks).

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. For example, the oil and gas industry use integrated control systems to manage refining operations at plant sites, remotely monitor the pressure and flow of gas pipelines, and control the flow and pathways of gas transmission. Water utilities can remotely monitor well levels and control the wells pumps; monitor flows, tank levels, or pressure in storage tanks; monitor pH, turbidity, and chlorine residual; and control the addition of chemicals to the water.

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 infrastructures, such as electric power distribution, oil and natural gas distribution, water and waste-water treatment, and transportation systems. They are also at the core of health-care devices, weapons systems, and transportation management. 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 decades, and yet, there have been few confirmed cases of cyberattacks. **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.**

No other attack demonstrates the threat to control systems as the

Stuxnet worm [1, 2]. The ultimate goal of Stuxnet is to sabotage that facility by reprogramming controllers to operate, most likely, out of their specified boundaries [1]. Stuxnet demonstrates that the motivation and capability exists for creating computer attacks capable to achieve military goals [3].

Not only can Stuxnet cause devastating consequences, but it is also very difficult to detect. Because Stuxnet used zero-day vulnerabilities, antivirus software would not have prevented the attack. In fact, the level of sophistication of the attack prevented some well known security companies such as Kaspersky to detect it initially [4]. In addition, victims attempting to detect modifications to their embedded controllers would not see any rogue code as Stuxnet hides its modifications with sophisticated PLC rootkits, and validated its drivers with trusted certificates.

The main motivation behind this work is the observation that while attackers may be able to hide the specific information technology methods used to exploit the system and reprogram their computers, they cannot hide their final goal: the need to cause an adverse effect on the physical system by sending malicious sensor or controller data that will not match the behavior expected by a supervisory control or an anomaly detection system.

Therefore, in this paper we explore security mechanisms that detect attacks by monitoring the physical system under control, and the sensor and actuator values. Our goal is to detect modifications to the sensed or controlled data as soon as possible, before the attack causes irreversible damages to the system (such as compromising safety margins).

In the rest of the paper we first summarize the vulnerability of control systems by discussing known attacks. We then discuss the efforts for securing control systems solely from an information technology perspective and identify the new and unique research problems that can be formulated by including a model of the physical system under control. We then develop a new attack detection algorithm and study the methodology on how to evaluate anomaly detection algorithms and their possible response strategies.

2. THE VULNERABILITY OF CONTROL SYSTEMS AND STUXNET

There have been many computer-based incidents in control systems. **Computer-based accidents** can be caused by any unanticipated software error, like the power plant shutdown caused by a computer rebooting after a patch [5]. **Non-targeted attacks** are incidents caused by the same attacks that any computer connected to the Internet may suffer, such as the Slammer worm infecting the Davis-Besse nuclear power plant [6], or the case of a controller being used to send spam in a water filtering plant [7].

However, the biggest threat to control systems are **Targeted attacks**. These attacks are the ones where the miscreants know that they are targeting control systems, and therefore, *they tailor their attack strategy with the aim of damaging the physical system under control*. Targeted attacks against control systems are not new. Physical attacks–for extortion and terrorism–are a reality in some countries [8]. Cyber-attacks are a natural progression to physical attacks: they are cheaper, less risky for the attacker, are not constrained by distance, and are easier to replicate and coordinate.

A classic computer-based targeted attack to SCADA systems is the attack on Maroochy Shire Council's sewage control system in Queensland, Australia [9]. There are many other reported targeted attacks [10–16]; however, no other attack has demonstrated the threats that control systems are subject to as well as the Stuxnet worm [1, 2]. Stuxnet has made clear that there are groups with the motivation and skills to mount sophisticated computer-based attacks to critical infrastructures, and that these attacks are not just speculations or belong only in Hollywood movies.

Stuxnet intercepts routines to read, write and locate blocks on a Programmable Logic Controller (PLC). By intercepting these requests, Stuxnet is able to modify the data sent to or returned from the PLC without the operator of the PLC ever realizing it [1].

Stuxnet was discovered on systems in Iran in June 2010 by researchers from Belarus–from the company VirusBlokAda; however, it is believed to have been released more than a year before. Stuxnet is a worm that spreads by infecting Windows computers. It uses multiple methods and zero-day exploits to spread itself via LANs or USB sticks. It is likely that propagation by LAN served as the first step, and propagation through removable drives was used to reach PCs not connected to other networks–therefore being isolated from the Internet or other networks is not a complete defense.

Once Stuxnet infects a Windows computer, it installs its own drivers. Because these drivers have to be signed, Stuxnet used two stolen certificates. Stuxnet also installs a rootkit to hide itself. The goal of the worm in a Windows computer is to search for WinCC/Step 7, a type of software used to program and monitor PLCs. (PLCs are the embedded systems attached to sensors and actuators that run control algorithms to keep the physical system operating correctly. They are typically programmed with a ladder logic program: a logic traditionally used to design control algorithms for panels of electromechanical relays.)

If Stuxnet does not find the WinCC/Step 7 software in the infected Windows machine, it does nothing; however, if it finds the software, it infects the PLC with another zero-day exploit, and then reprograms it. Stuxnet also attempts to hide the PLC changes with a PLC rootkit.

The reprogramming is done by changing only particular parts of the code–*overwriting certain process variables every five seconds and inserting rouge ladder logic*–therefore it is impossible to predict the effects of this change without knowing exactly how the PLC is originally programmed and what it is connected to, since the PLC program depends on the physical system under control, and typically, physical system parameters are unique to each individual facility. This means that the attackers were targeting a very specific PLC program and configuration (i.e., a very specific control system deployment).

Many security companies, including Symantec and Kaspersky have said that Stuxnet is the most sophisticated attack they have ever analyzed, and it is not difficult to see the reasons. Stuxnet uses four zero-day exploits, a Windows rootkit, the first known PLC rootkit, antivirus evasion techniques, peer-to-peer updates, and stolen certificates from trusted CAs. There is evidence that Stuxnet kept evolving since its initial deployment as attackers upgraded the infections with encryption and exploits, apparently adapting to conditions they found on the way to their target. The command and control architecture used two servers if the infected machines were able to access the Internet, or a peer to peer messaging system that could be used for machines that are offline. In addition, the attackers had a good level of intelligence about their target; they knew all the details of the control system configuration and its programs.

The sophistication of this attack has lead many to speculate that Stuxnet is the creation of a state-level sponsored attack. Since Iran has an unusually high percentage of the total number of reported infections of the worm in the world [1], there has been some speculation that their target was a specific industrial control system in Iran [2], such as a gas pipeline or power plant.

We argue that a threat like the Stuxnet worm must be dealt with defense-in-depth mechanisms like anomaly detection schemes. While traditional anomaly detection mechanisms may have some drawbacks like false alarms, we argue that for certain control systems, anomaly detection schemes focusing on the physical system–instead of using software or network models–can provide good detection capabilities with negligible false alarm rates.

3. NEW SECURITY PROBLEMS FOR CONTROL SYSTEMS

3.1 Efforts for Securing Control Systems

Most of the efforts for protecting control systems (and in particular SCADA) have focused on safety and reliability (the protection of the system against random and/or independent faults). Traditionally, control systems have not dealt with intentional actions or systematic failures. There is, however, an urgent growing concern for protecting control systems against malicious cyberattacks [6, 17–24].

There are several industrial and government-led efforts to improve the security of control systems. Several sectors–including chemical, oil and gas, and water–are currently developing programs for securing their infrastructure. The electric sector is leading the way with the North American Electric Reliability Corporation (NERC) cybersecurity standards for control systems [25]. NERC is authorized to enforce compliance to these standards, and it is expected that all electric utilities are fully compliant with these standards by the end of 2010.

NIST has also published a guideline for security best practices for general IT in Special Publication 800-53. Federal agencies must meet NIST SP800-53. To address the security of control systems, NIST has also published a Guide to Industrial Control System (ICS) Security [26], and a guideline to smart grid security in NIST-IR 7628. Although these recommendations are not enforceable, they can provide guidance for analyzing the security of most utility companies.

ISA (a society of industrial automation and control systems) is developing ISA-SP 99: a security standard to be used in manufacturing and general industrial controls.

The Department of Energy has also led security efforts by establishing the national SCADA test bed program [27] and by developing a 10-year outline for securing control systems in the energy sector [21]. The report–released in January 2006–identifies four main goals (in order from short-term goals to long-term goals): (1) measure the current security posture of the power grid, (2) develop and integrate protective measures, (3) implement attack detection and response strategies; and (4) sustain security improvements.

The use of wireless sensor networks in SCADA systems is becoming pervasive, and thus we also need to study their security. A number of companies have teamed up to bring sensor networks in the field of process control systems, and currently, there are two working groups to standardize their communications [28, 29]. Their wireless communication proposal has options to configure hop-by-hop and end-to-end confidentiality and integrity mechanisms. Similarly they provide the necessary protocols for access control and key management.

All these efforts have essentially three goals: (1) create awareness of security issues with control systems, (2) help control systems operators and IT security officers design a security policy, and (3) recommend basic security mechanisms for prevention (authentication, access controls, etc), detection, and response to security breaches.

While these recommendations and standards have placed significant importance on *survivability* of control systems (their ability to operate while they are under attack); we believe that they have not explored some new research problems that arise when control systems are under attack.

3.2 Differences

While it is clear that the security of control systems has become an active area in recent years, we believe that, so far, no one has been able to articulate what is new and fundamentally different in this field from a research point of view when compared to traditional IT security.

In this paper we would like to start this discussion by summarizing some previously identified differences and by proposing some new problems.

The property of control systems that is most commonly brought up as a distinction with IT security is that software **patching and frequent updates, are not well suited for control systems**. For example, upgrading a system may require months of advance in planning how to take the system offline; it is, therefore, economically difficult to justify suspending the operation of an industrial computer on a regular basis to install new security patches. Some security patches may even violate the certification of control systems, or–as previously mentioned–cause accidents to control systems [5].

Patching, however, is not a fundamental limitation to control systems. A number of companies have demonstrated that a careful antivirus and patching policy (e.g., the use of tiered approaches) can be used successfully [30, 31]. Also, most of the major control equipment vendors now offer guidance on both patch management and antivirus deployment for their control products. Thus there is little reason for SCADA system operators not to have good patch and antivirus programs in place today [32].

Large industrial control systems also have a large amount of **legacy systems**. Lightweight cryptographic mechanisms to ensure data integrity and confidentiality have been proposed to secure these systems [33, 34]. The recent IEEE P1711 standard is designed for providing security in legacy serial links [35]. Having some small level of security is better than having no security at all; however, *we believe that most of the efforts done for legacy systems should be considered as short-term solutions*. For properly securing critical control systems the underlying technology must satisfy some minimum performance requirements to allow the implementation of well tested security mechanisms and standards.

Another property of control systems that is commonly mentioned is the real-time requirements of control systems. Control systems are autonomous decision making agents which need to make decisions in real time. While availability is a well studied problem in information security, **real-time availability** provides a stricter operational environment than most traditional IT systems. We show in this paper that real-time availability requirements depend on the dynamics (fast vs. slow) of the physical system.

Not all operational differences are more severe in control systems than in traditional IT systems. By comparison to enterprise systems, control systems exhibit comparatively **simpler network dynamics**: Servers change rarely, there is a fixed topology, a stable user population, regular communication patterns, and a limited number of protocols. Therefore, implementing network intrusion detection systems, anomaly detection, and white listing may be easier than in traditional enterprise systems [36].

3.3 What is new and fundamentally different?

While all these differences are important, we believe that the major distinction of control systems with respect to other IT systems is the interaction of the control system with the physical world.

While current tools from information security can give *necessary* mechanisms for securing control systems, these mechanisms alone are not *sufficient* for defense-in-depth of control systems. When attackers bypass basic security defenses they may be able to affect the physical world.

In particular, research in computer security has focused traditionally on the protection of information; but it has not considered how attacks affect *estimation* and *control* algorithms–and ultimately, how attacks affect the physical world.

We believe that by understanding the interactions of the control system with the physical world, we should be able to develop a general and systematic framework for securing control systems in three fundamentally new areas:

1. Better understand the consequences of an attack for *risk assessment*. While there has been previous risk assessment studies on cyber security for SCADA systems [18, 37–39], currently, there are few studies on identifying the attack strategy of an adversary, once it has obtained unauthorized access to some control network devices. Notable exceptions are the study of false data-injection attacks to state estimation in power grids [40–45], and electricity markets [46]. We need further research to understand the threat model in order to design appropriate defenses and to invest in securing the most critical sensors or actuators.

2. Design new attack-detection algorithms. By monitoring the behavior of the physical system under control, we should be able to detect a wide range of attacks by compromised measurements. The work closest to ours are the study of false data injection attacks in control systems [47] and the intrusion detection models of Rrushi [48]–this last work; however, does not consider *dynamical models* of the process control system. We need further research on dynamical system models used in control theory as a tool for specification-based intrusion detection systems.

3. Design new attack-resilient algorithms and architectures: we need to design and operate control systems *to survive* an intentional cyber assault with no loss of critical functions. Our goal is to design systems where even if attackers manage to bypass some basic security mechanisms, they will still face several control-specific security devices that will minimize the damage done to the system. In particular, we need to investigate how to reconfigure and adapt control systems when they are under an attack to increase the resiliency of the system. We are not aware of any other work on designing new control algorithms or reconfiguration and control algorithms able to withstand attacks, or that reconfigure their operations based on detected attacks. There is previous work on safety and fault diagnosis; however, as we explain in this paper, these systems are not enough for detecting deception attacks launched by an intelligent attacker with knowledge on how to evade fault detection methods used by the system.

In the next sections we describe our ideas, experiments, and results for (1) risk-assessment, (2) false-data-injection detection, and (3) automatic attack-response in process control systems. In each section we first present a general theory for approaching the topic, and then for experimental validation, we implement our ideas to the model of a chemical reactor process.

4. RISK ASSESSMENT

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 $R_\mu = \mathbb{E}[L] \approx \sum_i L_i p_i$, 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 $R_\chi = \mathbb{E}[L^2] - R_\mu$ 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 section 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.

4.1 Attack models

We consider the case when the state of the system is measured by a sensor network of p sensors with measurement vector $y(k) = \{y_1(k), \ldots, y_p(k)\}$, 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 $\forall k, y_i(k) \in [y_i^{min}, y_i^{max}]$. Let $\mathcal{Y}_i = [y_i^{min}, y_i^{max}]$. We assume each sensor has a unique identity protected by a cryptographic key.

Let $\tilde{y}(k) \in \mathbb{R}^p$ 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 $\mathcal{Y}_i$ (signals outside this range can be easily detected by fault-tolerant algorithms).

Let $\mathcal{K}_a = \{k_s, \ldots, k_e\}$ 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:

$$\tilde{y}_i(k) = \begin{cases} y_i(k) & \text{for } k \notin \mathcal{K}_a \\ a_i(k) & \text{for } k \in \mathcal{K}_a,\, a_i(k) \in \mathcal{Y}_i \end{cases}$$

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.

4.2 Experiments

To test our attacks, we use the Tennessee-Eastman process control system (TE-PCS) model and the associated multi-loop PI control law as proposed by Ricker [49]. We briefly describe the process architecture and the control loops in Figure 1. The original process model is implemented in FORTRAN and the PI control law is implemented in MATLAB. We use this code for our study.

The chemical process consists of an irreversible reaction which occurs in the vapour phase inside a reactor of fixed volume V of 122 (m^3). Two non-condensible reactants A and C react in the

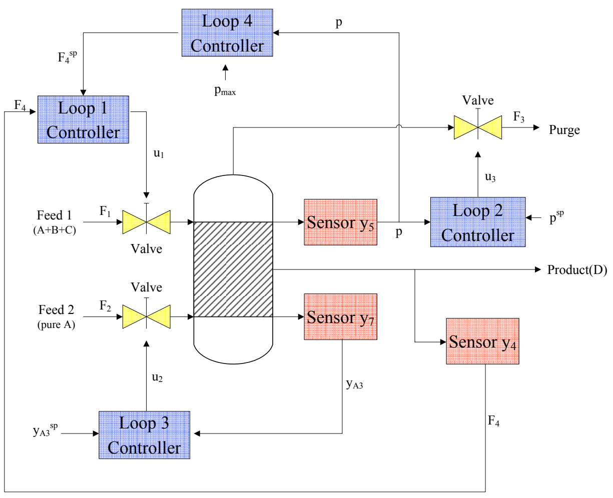


Figure 1: Architecture of the Simplified TE Plant.

presence of an inert B to form a non-volatile liquid product D:

$$A + C \xrightarrow{B} D.$$

The feed stream 1 contains A, C and trace of B; feed stream 2 is pure A; stream 3 is the purge containing vapours of A, B, C; and stream 4 is the exit for liquid product D. The measured flow rates of stream i is denoted by F_i $(kmol\ h^{-1})$. The *control objectives* are

- *Regulate* F_4, the rate of production of the product D, at a set-point F_4^{sp} $(kmol\ h^{-1})$,
- Maintain P, the operating pressure of the reactor, below the shut-down limit of 3000 kPa as dictated *safety* considerations,
- Minimize C, the *operating cost* measured in (kmol-of-product). The cost depends linearly on the purge loss of A and C relative to the production rate of D. The cost considerations dictate that the pressure be maintained as close as possible to 3000 kPa.

The production rate of D, denoted by r_D $(kmol\ h^{-1})$ is

$$r_D = k_0 y_{A3}^{v1} y_{C3}^{v2} P^{v3},$$

where y_{A3} and y_{C3} denote the respective fractions of A and C in the purge and v_1, v_2, v_3 are given constants.

There are four *input variables* (or command signals) available to achieve the above control objectives. The first three input variables, denoted as u_1, u_2 and u_3, trigger the actuators that can change the positions of the respective valves. The fourth input variable, denoted as u_4, is the set point for the proportional controller for the liquid inventory. The input variables as used by the controller in the following way:

- Production rate $y_4 = F_4$ is controlled using Feed 1 (u_1) by loop-1 controller,
- Pressure $y_5 = P$ is controlled using the purge rate (u_3) by loop-2 controller,
- Partial pressure of product A in the purge $y_7 = y_{A3}$ is controlled using Feed 2 (u_3) by loop-3 controller,

When u_3 saturates, the loop-4 controller uses u_1 to control the pressure P. The controllers for all four loops in figure 1 are *proportional integral* (PI) controllers.

In steady-state operation, the production rate F_4 is 100 $kmol\ h^{-1}$, the pressure P is 2700 KPa and the fraction of A in the purge is 47 $mol\%$.

We study the security issues of control systems by experimenting and simulating cyber attacks on sensor signals in the TE-PCS model. Because operating the chemical reactor with a pressure larger than 3000 kPa is unsafe (it may lead to an explosion or damage of the equipment) We.assume that that the goal of the attacker is to raise the pressure level of the tank to a value larger than 3000 kPa. We model an attacker that only has access to a single sensor at a given time. We also assume $L_i > L_j$, when an attack i can drive the system to an unsafe state and an attack j cannot, and $L_i = L_j$ if both attacks i and j either do not drive the system to an unsafe state, or both can compromise the safety of the sytem.

From the experimental results, we found that the most effective of these attacks were max/min attacks (i.e., when $a_i(k) = y_i^{min}$ or $a_i(k) = y_j^{max}$). However, not all of the max/min attacks were able to drive the pressure to unsafe levels. We now summarize some of the results.

- By attacking the sensors, a controller is expected to respond with incorrect control signals since it receives wrong information from the compromised sensors. For example, by forging y_7 as y_7^{max} from $t = 0$ to 30, the controller believes there is a large amount of component A in the tank.

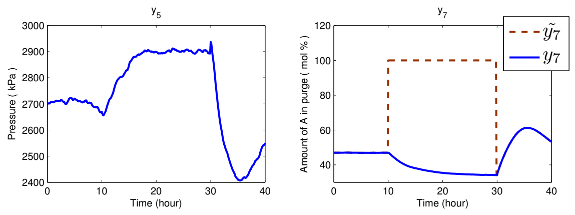
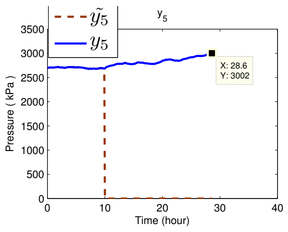


Figure 2: Integrity attack y_7^{max} from $t = 0$ to 30. The system remains in a safe state for attacks on y_7.

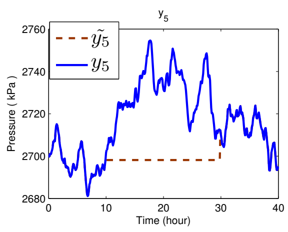
From the experiments, we found that the plant system can go back to the steady state after the attack finishes, as illustrated in Fig 2. Furthermore, the pressure in the main tank never reaches 3000 kPa. In general we found that the plant is very resilient to attacks on y_7 and y_4. Attacks in the limit of the sensing range (y^{min} and y^{max}) were the more damaging, but they did not force the system into an unsafe state.

- By launching attack y_5^{min} the controller turns down the purge valve to increase the pressure and prevent the liquid products from accumulating. We can see that the real pressure of the tank (y_5 in Fig 3(a)) keeps increasing past 3000 kPa and the system operates in an unsafe state. In this experiment, it takes about 20 hours ($t = 10$ to $t = 30$) to shut down (or cause an explosion to) the plant. This long delay in causing an effective attack may give defenders the advantage: for physical processes with *slow-dynamics*, it is possible that human system operators may have enough time to observe unusual phenomenon and take proper actions against the attack.

- We found out that in general DoS attacks do not affect the plant. We ran the plant 20 times for 40 hours each and for a DoS attack lasting 20 hours the pressure in the tank never exceeded 2900kPa.



9(a)



9(b)

Figure 3: Safety can be breached by compromising sensor y_5 (3(a)). DoS attacks, on the other hand, do not cause any damage (and they are easy to detect.) (3(b)).

We conclude that if the plant operator wants to prevent an attack from making the system operate in an unsafe state, it should prioritize defenses against integrity attacks rather than on DoS attacks. If the plant operator only has enough budget to deploy advanced security mechanisms for one sensor (e.g., tamper resistance, or TPM chips), y_5 should be the priority.

5. DETECTION OF ATTACKS

Detecting attacks to control systems can be formulated as an anomaly-based intrusion detection problem [50]. 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. We revisit this problem in the next section.

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 5.1 we discuss our choice of linear models as an approximation of the behavior of the physical system. In section 5.2, we describe change detection theory and the detection algorithm we use–a nonparametric cumulative sum (CUSUM) statistic.

5.1 Linear Model

To develop accurate control algorithms, control engineers often construct a representative model that captures the behavior of the physical system in order to predict how the system will react to a given control signal. A process model can be derived from first principles (a model based on the fundamental laws of physics) or from empirical input and output data (a model obtained by simulating the process inputs with a carefully designed test sequence). It is also very common to use a combination of these two models; for example, first-principle models are typically calibrated by using process test data to estimate key parameters. Likewise, empirical models are often adjusted to account for known process physics [51, 52].

For highly safety-critical applications, such as the aerospace industry, it is technically and economically feasible to develop accurate models from first principles [51]. However, for the majority of process control systems, the development of process models from fundamental physics is difficult.

In many cases such detailed models are difficult to justify economically, and even impossible to obtain in reasonable time due to the complex nature of many systems and processes. (The TE-PCS system used in our experiments is one of the few cases available in the literature of a detailed nonlinear model of an industrial control problem; this is the reason why the TE-PCS system has been used as a standard testbed in many industrial control papers.)

To facilitate the creation of physical models, most industrial control vendors provide tools (called identification packages) to develop models of physical systems from training data. The most common models are *linear* systems. Linear systems can be used to model dynamics that are linear in state $x(k)$ and control input $u(k)$

$$x(k+1) = Ax(k) + Bu(k) \tag{1}$$

where time is represented by $k \in \mathbb{Z}^+$, $x(k) = (x_1(k), \ldots, x_n(k)) \in \mathbb{R}^n$ is the state of the system, and $u(k) = (u_1(k), \ldots, u_m(k)) \in \mathbb{R}^m$ is the control input. The matrix $A = (a_{ij}) \in \mathbb{R}^{n\times n}$ models the physical dependence of state i on state j, and $B = (b_{ij}) \in \mathbb{R}^{n\times m}$ is the input matrix for state i from control input j.

Assume the system (1) is monitored by a *sensor network* with p sensors. We obtain the measurement sequence from the observation equations

$$\hat{y}(k) = Cx(k), \tag{2}$$

where $\hat{y}(k) = (\hat{y}_1(k), \ldots, \hat{y}_p(k)) \in \mathbb{R}^p$, and $\hat{y}_l(k) \in \mathbb{R}$ is the estimated measurement collected by sensor l at time k. Matrix $C \in \mathbb{R}^{p\times n}$ is called output matrix.

5.2 Detection Methods

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 [53].

In optimal stopping problems, we are given a time series sequence $z(1), z(2), \ldots, z(N)$, 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 k_s 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.

5.2.1 *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 [54] (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 [55], worms [56], proxies used by spammers [57], and botnets [58].

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:

$$S(k+1) = \log \frac{p_1(z(k))}{p_0(z(k))} + S(k)$$
$$N = \inf_n \{n : S(n) \notin [L, U]\},$$

starting with $S(0) = 0$. The SPRT decision rule d_N is defined as:

$$d_N = \begin{cases} H_1 & \text{if } S(N) \geq U \\ H_0 & \text{if } S(N) \leq L, \end{cases} \tag{3}$$

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).

5.2.2 *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 $N \geq k_s$) for which the test stops and decides that a change has occurred. Let $S(0) = 0$. The CUSUM statistic is updated according to

$$S(k+1) = \left(\log \frac{p_1(z(k))}{p_0(z(k))} + S(k)\right)^+ \tag{4}$$

where $(a)^+ = a$ if $a \geq 0$ and zero otherwise. The stopping time is:

$$N = \inf_n \{n : S(n) \geq \tau\} \tag{5}$$

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 re-starts.

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

$$z_i(k) := \|\tilde{y}_i(k) - \hat{y}_i(k)\| - b_i \tag{6}$$

where b_i is a small positive constant chosen such that

$$\mathbb{E}_0[\|\tilde{y}_i(k) - \hat{y}_i(k)\| - b_i] < 0. \tag{7}$$

The nonparametric CUSUM statistic for sensor i is then:

$$S_i(k) = (S_i(k-1) + z_i(k))^+, S_i(0) = 0 \tag{8}$$

and the corresponding decision rule is

$$d_{N,i} \equiv d_\tau(S_i(k)) = \begin{cases} H_1 & \text{if } S_i(k) > \tau_i \\ H_0 & \text{otherwise.} \end{cases} \tag{9}$$

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

Following [59], we state the following two important results for Eq. (8)-(9):

- 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.

5.3 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.

5.4 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:

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

The resulting attack (for y_5 and y_4) is:

$$\tilde{y}_i(k) = \begin{cases} y_i^{min} & \text{if } S_i(k+1) \leq \tau_i \\ \hat{y}_i(k) - |\tau_i + b_i - S_i(k)| & \text{if } S_i(k+1) > \tau_i \end{cases}$$

For y_7 we use

$$\tilde{y}_7(k) = \begin{cases} y_7^{max} & \text{if } S_{y_7}(k) \leq \tau_7 \\ \hat{y}_7 + |\tau_7 + b_7 - S_{y_7}(k)| & \text{if } S_{y_7}(k) > \tau_7 \end{cases}$$

5.5 Bias Attacks

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

$$\tilde{y}_{i,k} = \hat{y}_{i,k} - c_i \in \mathcal{Y}_i$$

In this case, the nonparametric CUSUM statistic can be written as:

$$S_i(n) = \sum_{k=0}^{n-1} |\hat{y}_i(k) - \tilde{y}_i(k)| - nb_i$$

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:

$$\sum_{k=0}^{n-1} c_i = \tau_i + nb_i$$

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 $\tilde{y}_i \in \mathcal{Y}_i$ 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.

5.6 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:

$$\tilde{y}_i(k) = \hat{y}_i(k) - \beta_i \alpha_i^{n-k}.$$

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.

$$\sum_{k=0}^{n-1} \beta_i \alpha_i^{n-k} - nb_i = \tau_i$$

This addition is a geometric progression.

$$\sum_{k=0}^{n-1} \beta_i \alpha_i^{n-k} = \beta_i \alpha_i^n \sum_{k=0}^{n-1} (\alpha_i^{-1})^k = \beta_i \frac{1 - \alpha_i^n}{\alpha_i^{-1} - 1}$$

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

5.7 Experiments

We continue our use of the TE-PCS model. In this section we first describe our selection criteria for matrices A, B, and C for the linear model, and the parameters b_i and τ_i for the CUSUM statistic. We then describe the tradeoffs between false alarm rates and the delay for detecting attacks. The section ends with the study of stealthy attacks.

5.7.1 Linear Model

In this paper we use the linear system characterized by the matrices A, B, and C, obtained by linearizing the non-linear TE-PCS model about the steady-state operating conditions. (See Ricker [49].) The linear model is a good representative of the actual TE-PCS model when the operating conditions are reasonably close to the steady-state.

5.7.2 Nonparametric CUSUM parameters

In order to select b_i for each sensor i, we need to estimate the expected value of the distance $|\hat{y}_i(k) - y_i(k)|$ between the linear model estimate $\hat{y}_i(k)$ and the sensor measurement $y_i(k)$ (i.e., the sensor signal without attacks).

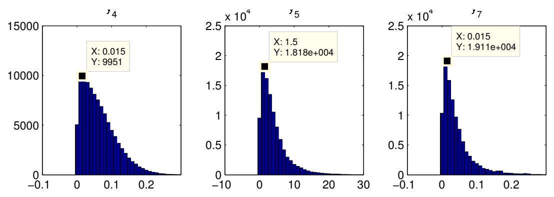


Figure 4: The paramenter of ADM: b. For y_4, 9951 bs are 0.015. The mean value of b_{y_4} is 0.0642.

We run experiments for ten thousand times (and for 40 hours each time) without any attacks to gather statistics. Fig 4 shows the estimated probability distributions (without normalization).

To obtain b_i, we compute the empirical expected value for each distance and then round up to the two most significant units. We obtain $b_{y_4} = 0.065$, $b_{y_5} = 4.1$, $b_{y_7} = 0.042$.

Once we have b_i for each sensor, we need to find a threshold τ_i to balance the tradeoff between false alarms and detection time.

False Alarm Rate.

We run simulations for twenty times without attacks and compute the total number of false alarms for different values of τ (and for each sensor). Fig 5 shows the results. Taking y_4 as an example, we notice that S_{y_4} alerts frequently if we set $\tau_{y_4} < 6$.

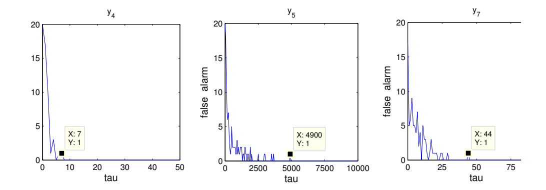


Figure 5: The number of false alarms decreases exponentially with increasing τ. This results confirm the theory supporting the nonparametric CUSUM algorithm.

In general, we would like to select τ as high as possible for each sensor to avoid any false alarm; however, increasing τ increases the time to detect attacks.

Detection Time.

To measure the time to detect attacks, we run simulations by launching scaling attacks ($a_i(k) = \lambda_m y_i(k)$) on sensors y_4, y_5 and y_7. Figs 6 and 7 shows the experimental results.

The selection of τ is a trade-off between detection time and the number of false alarms. The appropriate value differs from system to system. Because the large number of false alarms is one of the main problems for anomaly detection systems, and because the TE-PCS process takes at least 10 hours to reach the unsafe state (based on our risk assessment section), we choose the conservative set of parameters $\tau_{y_4} = 50$, $\tau_{y_5} = 10000$, $\tau_{y_7} = 200$. These parameters allow us to detect attacks within a couple of hours, while not raising any false alarms.

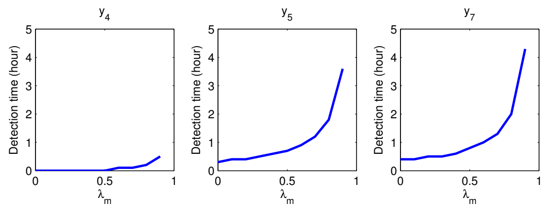


Figure 6: Detection time v.s. scaling attack. Note that for $\lambda_i^m = 1$ there is no alarm.

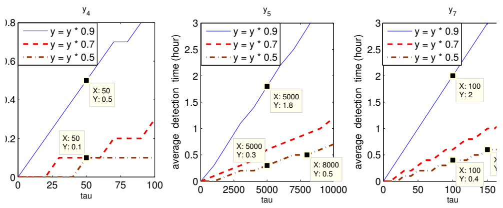


Figure 7: The time for detection increases linearly with increasing τ. This results confirm the theory behind the nonparametric CUSUM algorithm.

5.7.3 *Stealthy Attacks*

To test if our selected values for τ are resilient to stealthy attacks, we decided to investigate the effect of stealhty attacks as a function of τ. To test how the attacks change for all thresholds we parameterize each threshold by a parameter p: $\tau_i^{test} = p\tau_i$. Fig. 8 shows the percentage of times that geometric stealthy attacks (assuming the attacker controls all three sensor readings) were able to drive the pressure above 3000kPa while remaining undetected (as a function of p).

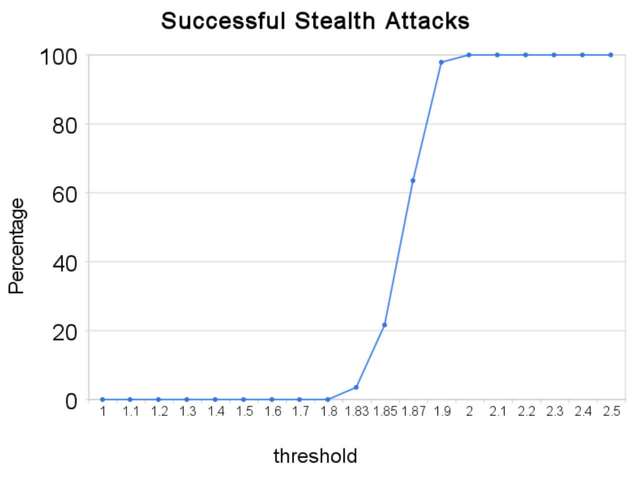


Figure 8: Percentage of stealthy attacks that increase the pressure of the tank above 3,000kPa as a function of scaling parameter p.

We implemented all stealth attacks starting at time $T = 10$ (hrs). We assume the goal of the attacker is to be undetected until $T = 30$ (hrs). For example, Fig. 9 shows the results of attacking all three sensors with a geometric attack. The nonparametric CUSUM statistic shown in Fig. 10 shows how the attacker remains undetected until time $T = 30$ (hrs).

We found that a surge attack does not cause significant damages because of the inertia of the chemical reactor: by the time the statistic reaches the threshold τ, the chemical reactor is only starting to respond to the attack. However, since the attacker can only add very small variations to the signal once it is close to the threshold, the attack ceases to produce any effect and the plant continues operating normally.

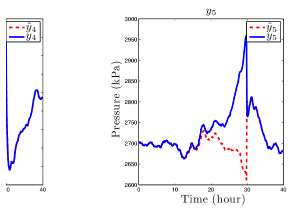


Figure 9: Geometric attacks to the three 3 sensors. The solid lines represent the real state of the system, while the dotted lines represent the information sent by the attacker.

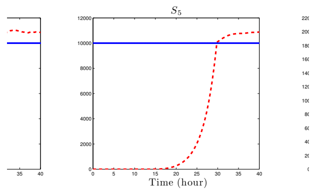


Figure 10: Statistics of geometric attacks with 3 sensors compromised.

Finally, we assume two types of attackers. An attacker that has compromised y_5 (but who does not know the values of the other sensors, and therefore can only control $S_{y_5}(k)$), and an attacker that has compromised all three sensors (and therefore can control the statistic $S(k)$ for all sensors). We launched each attack 20 times. The results are summarized in Figure 11.

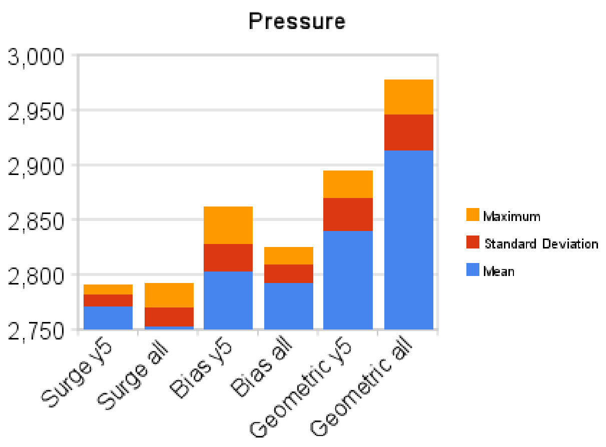


Figure 11: Effect of stealthy attacks. Each attack last 20 hours.

Our results show that even though our detection algorithm fails to detect stealthy attacks, we can keep the the plant in safe conditions. We also find that the most successful attack strategy are geometric attacks.

6. 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. For example, many web servers send CAPTCHAs to the client whenever they find that connections resemble bot connections, firewalls drop connections that conform to their rules, the execution of anomalous processes can be slowed down by intrusion detection systems, etc.

Given that we already have an estimate for the state of the system (given by a linear model), a natural response strategy for control systems is to use this estimate when the anomaly detection statistic fires an alarm. Fig 12 shows our proposed architecture. Specifically: for sensor i, if $S_i(k) > \tau_i$, the ADM replaces the sensor measurements $\tilde{y}_i(k)$ with measurements generated by the linear model $\hat{y}_i(k)$ (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.

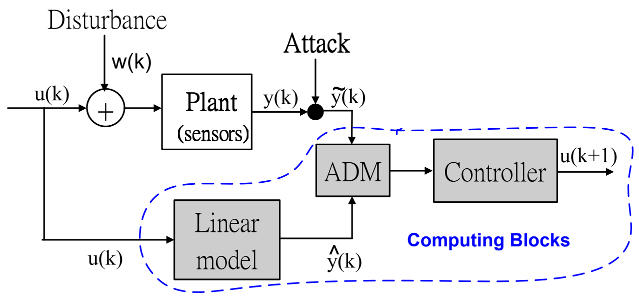


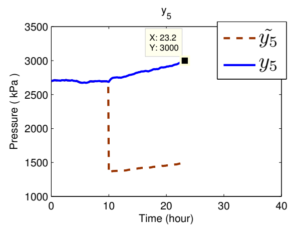
Figure 12: An Anomaly Detection Module (ADM) can detect an attack and send an estimate of the state of the system to the controller.

Introducing automatic response mechanisms is, however, not an easy solution. Every time systems introduce an automatic response to an alarm, they have to consider the cost of dealing with false alarms. In our proposed detection and response architecture (Fig. 12), we have to make sure that if there is a false alarm, controlling the system by using the estimated values from the linear system will not cause any safety concerns.

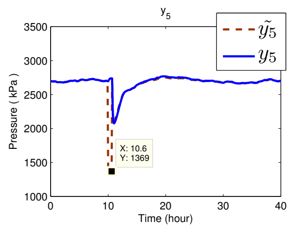
6.1 Experiments

The automatic response mechanism works well when we are under attack. For example, Fig. (13) 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 1000 times; each



9(a) Without ADM



9(b) ADM detects and responds to the attack at $T = 10.7$ (hr)

Figure 13: $\tilde{y_5} = y_5 * 0.5$

Alarms	Avg y_5	Std Dev	Max y_5
0	2700.4	14.73	2757

Table 1: For Thresholds $\tau_{y_4} = 50, \tau_{y_5} = 10000, \tau_{y_7} = 200$ we obtain no false alarm. Therefore we only report the expected pressure, the standard deviation of the pressure, and the maximum pressure reached under no false alarm.

time the experiment ran for 40 hours. As expected, with the parameter set $\tau_{y_4} = 50$, $\tau_{y_5} = 10000$, $\tau_{y_7} = 200$ our system did not detect any false alarm (see Table 1); therefore we decided to reduce the detection threshold to $\tau_{y_4} = 5$, $\tau_{y_5} = 1000$, $\tau_{y_7} = 20$ and run the same experiments again. Table 2 shows the behavior of the pressure *after the response to a false alarm.* 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 $2779 kPa$, which is in the same order of magnitude than the normal variation of the pressure without any false alarm ($2757 kPa$).

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 (a couple of hours).

7. CONCLUSIONS

In this work we identified three new 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 also studied the use of physical models for anomaly detection and proposed three generic types of stealthy attacks. Finally, we proposed the use of automatic response mechanisms based on estimates of the state 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

	Alarms	Avg y_5	Std Dev	Max y_5
y_4	61	2710	30.36	2779
y_5	106	2705	18.72	2794
y_7	53	2706	20.89	2776

Table 2: Behavior of the plant after response to a false alarm with thresholds $\tau_{y_4} = 5, \tau_{y_5} = 1000, \tau_{y_7} = 20$.

responding to a false alarm.

In our experiments with the TE-PCS process we found several interesting results. (1) Protecting against integrity attacks is more important than protecting against DoS attacks. In fact, we believe that DoS attacks have negligible impact to the TE-PCS process. (2) The chemical reactor process is a well-behaved system, in the sense that even under perturbations, the response of the system follows very closely our linear models. In addition, the slow dynamics of this process allows us to be able to detect attacks even with large delays with the benefit of not raising any false alarms. (3) Even when we configure the system to have false alarms, we saw that the automatic response mechanism was able to control the system in a safe mode.

One of our main conclusions regarding the TE-PCS plant, is that it is a very resiliently-designed process control system. Design of resilient process control systems takes control system design experience and expertise. The design process is based on iteratively evaluating the performance on a set of bad situations that can arise during the operation of the plant and modifying control loop structures to build in resilience. In particular, Ricker's paper discusses the set of random faults that the four loop PI control is able to withstand.

We like to make two points in this regard: (1). The PI control loop structure is distributed, in the sense that no PI control loop controls all actuators and no PI loop has access to all sensor measurements, and (2). The set of bad situations to which this control structure is able to withstand may itself result from the one or more cyber attacks. However, even though the resilience of TE-PCS plant is ensured by expert design, we find it interesting to directly test this resilience within the framework of assessment, detection and response that we present in this article.

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).

Even though we have focused on the analysis of a chemical reactor system, our principles and techniques can be applied to many other physical processes. An automatic detection and response module may not be a practical solution for all control system processes; however, we believe that many processes with similar characteristics to the TE-PCS can benefit from this kind of response.

Acknowledgments

We would like to thank Gabor Karsai, Adrian Perrig, Bruno Sinopoli, and Jon Wiley for helpful discussions on the security of control systems. This work was supported in part by by the iCAST-TRUST collaboration project, and by CHESS at UC Berkeley, which receives support from the NSF awards #0720882 (CSR-EHS: PRET) and #0931843 (ActionWebs), ARO #W911NF-07-2-0019, MURI #FA9550-06-0312, AFRL, and MuSyC.

8. REFERENCES

[1] Nicolas Falliere, Liam O Murchu, and Eric Chien. *W32.Stuxnet Dossier*. Symantec, version 1.3 edition, November 2010.

[2] Ralph Langner. Langner communications. `http://www.langner.com/en/`, October 2010.

[3] Steve Bellovin. Stuxnet: The first weaponized software? `http://www.cs.columbia.edu/~smb/blog//2010-09-27.html`, October 2010.

[4] Dale Peterson. Digital bond: Weisscon and stuxnet. `http://www.digitalbond.com/index.php/2010/09/22/weisscon-and-stuxnet/`, October 2010.

[5] Brian Krebs. *Cyber Incident Blamed for Nuclear Power Plant Shutdown*. Washington Post, `http://www.washingtonpost.com/wp-dyn/content/article/2008/06/05/AR2008060501958.html`, June 2008.

[6] Robert J. Turk. Cyber incidents involving control systems. Technical Report INL/EXT-05-00671, Idao National Laboratory, October 2005.

[7] Richard Esposito. Hackers penetrate water system computers. `http://blogs.abcnews.com/theblotter/2006/10/hackers_penetra.html`, October 2006.

[8] BBC News. *Colombia Rebels Blast Power Pylons*. BBC, `http://news.bbc.co.uk/2/hi/americas/607782.stm`, January 2000.

[9] Jill Slay and Michael Miller. Lessons learned from the maroochy water breach. In *Critical Infrastructure Protection*, volume 253/2007, pages 73–82. Springer Boston, November 2007.

[10] Paul Quinn-Judge. Cracks in the system. *TIME Magazine*, 9th Jan 2002.

[11] Thomas Reed. *At the Abyss: An Insider's History of the Cold War*. Presidio Press, March 2004.

[12] United States Attorney, Eastern District of California. Willows man arrested for hacking into Tehama Colusa Canal Authority computer system. `http://www.usdoj.gov/usao/cae/press_releases/docs/2007/11-28-07KeehnInd.pdf`, November 2007.

[13] United States Attorney, Eastern District of California. Sacramento man pleads guilty to attempting ot shut down california's power grid. `http://www.usdoj.gov/usao/cae/press_releases/docs/2007/12-14-07DenisonPlea.pdf`, November 2007.

[14] David Kravets. Feds: Hacker disabled offshore oil platform leak-detection system. `http://www.wired.com/threatlevel/2009/03/feds-hacker-dis/`, March 2009.

[15] John Leyden. Polish teen derails tram after hacking train network. *The Register*, 11th Jan 2008.

[16] Andrew Greenberg. Hackers cut cities' power. In *Forbes*, Jaunuary 2008.

[17] V.M. Igure, S.A. Laughter, and R.D. Williams. Security issues in SCADA networks. *Computers & Security*, 25(7):498–506, 2006.

[18] P. Oman, E. Schweitzer, and D. Frincke. Concerns about intrusions into remotely accessible substation controllers and SCADA systems. In *Proceedings of the Twenty-Seventh Annual Western Protective Relay Conference*, volume 160. Citeseer, 2000.

[19] US-CERT. *Control Systems Security Program*. US Department of Homeland Security, `http://www.us-cert.gov/control_systems/index.html`, 2008.

[20] GAO. Critical infrastructure protection. Multiple efforts to secure control systems are under way, but challenges remain. Technical Report GAO-07-1036, Report to Congressional Requesters, September 2007.

[21] Jack Eisenhauer, Paget Donnelly, Mark Ellis, and Michael O'Brien. *Roadmap to Secure Control Systems in the Energy Sector*. Energetics Incorporated. Sponsored by the U.S. Department of Energy and the U.S. Department of

Homeland Security, January 2006.

[22] Eric Byres and Justin Lowe. The myths and facts behind cyber security risks for industrial control systems. In *Proceedings of the VDE Congress, VDE Association for Electrical Electronic & Information Technologies*, October 2004.

[23] D. Geer. Security of critical control systems sparks concern. *Computer*, 39(1):20–23, Jan. 2006.

[24] A.A. Cardenas, T. Roosta, and S. Sastry. Rethinking security properties, threat models, and the design space in sensor networks: A case study in SCADA systems. *Ad Hoc Networks*, 2009.

[25] NERC-CIP. *Critical Infrastructure Protection*. North American Electric Reliability Corporation, `http://www.nerc.com/cip.html`, 2008.

[26] K. Stouffer, J. Falco, and K. Kent. Guide to supervisory control and data acquisition (SCADA) and industrial control systems security. Sp800-82, NIST, September 2006.

[27] Idaho National Laboratory. National SCADA Test Bed Program. `http://www.inl.gov/scada`.

[28] Hart. `http://www.hartcomm2.org/frontpage/wirelesshart.html`. *WirelessHart whitepaper*, 2007.

[29] ISA. `http://isa.org/isasp100`. *Wireless Systems for Automation*, 2007.

[30] Eric Cosman. Patch management at Dow chemical. In *ARC Tenth Annual Forum on Manufacturing*, February 20-24 2006.

[31] Patch management strategies for the electric sector. Edison Electric Institute–IT Security Working Group, March 2004.

[32] Eric Byres, David Leversage, and Nate Kube. Security incidents and trends in SCADA and process industries. *The Industrial Ethernet Book*, 39(2):12–20, May 2007.

[33] Andrew K. Wright, John A. Kinast, and Joe McCarty. Low-latency cryptographic protection for SCADA communications. In *Applied Cryptography and Network Security (ACNS)*, pages 263–277, 2004.

[34] Patrick P. Tsang and Sean W. Smith. YASIR: A low-latency high-integrity security retrofit for lecacy SCADA systems. In *23rd International Information Security Conference (IFIC SEC)*, pages 445–459, September 2008.

[35] Steven Hurd, Rhett Smith, and Garrett Leischner. Tutorial: Security in electric utility control systems. In *61st Annual Conference for Protective Relay Engineers*, pages 304–309, April 2008.

[36] Steven Cheung, Bruno Dutertre, Martin Fong, Ulf Lindqvist, Keith Skinner, and Alfonso Valdes. Using model-based intrusion detection for SCADA networks. In *Proceedings of the SCADA Security Scientific Symposium*, Miami Beach, FL, USA, 2007 2007.

[37] PAS Ralston, JH Graham, and JL Hieb. Cyber security risk assessment for SCADA and DCS networks. *ISA transactions*, 46(4):583–594, 2007.

[38] P.A. Craig, J. Mortensen, and J.E. Dagle. Metrics for the National SCADA Test Bed Program. Technical report, PNNL-18031, Pacific Northwest National Laboratory (PNNL), Richland, WA (US), 2008.

[39] G. Hamoud, R.L. Chen, and I. Bradley. Risk assessment of power systems SCADA. In *IEEE Power Engineering Society General Meeting, 2003*, volume 2, 2003.

[40] Yao Liu, Michael K. Reiter, and Peng Ning. False data injection attacks against state estimation in electric power grids. In *CCS '09: Proceedings of the 16th ACM conference on Computer and communications security*, pages 21–32, New York, NY, USA, 2009. ACM.

[41] Rakesh Bobba, Katherine M. Rogers, Qiyan Wang, Himanshu Khurana, Klara Nahrstedt, and Thomas J. Overbye. Detecting false data injection attacks on dc state estimation. In *Preprints of the 1st Workshop on Secure Control Systems*, 2010.

[42] Henrik Sandberg, Teixeira Andre, and Karl H. Johansson. On security indices for state estimators in power networks. In *Preprints of the 1st Workshop on Secure Control Systems*, 2010.

[43] Oliver Kosut, Liyan Jia, Robert J. Thomas, and Lang Tong. Malicious data attacks on smart grid state estimation: Attack strategies and countermeasures. In *First International Conference on Smart Grid Communications (SmartGridComm)*, pages 220–225, 2010.

[44] Oliver Kosut, Liyan Jia, Robert J. Thomas, and Lang Tong. On malicious data attacks on power system state estimation. In *UPEC*, 2010.

[45] A Teixeira, S. Amin, H. Sandberg, K.H. Johansson, and S.S. Sastry. Cyber-security analysis of state estimators in electric power systems. In *IEEE Conference on Decision and Control (CDC)*, 2010.

[46] Le Xie, Yilin Mo, and Bruno Sinopoli. False data injection attacks in electricity markets. In *First International Conference on Smart Grid Communications (SmartGridComm)*, pages 226–231, 2010.

[47] Yilin Mo and Bruno Sinopoli. False data injection attacks in control systems. In *Preprints of the 1st Workshop on Secure Control Systems*, 2010.

[48] Julian Rrushi. *Composite Intrusion Detection in Process Control Networks*. PhD thesis, Universita Degli Studi Di Milano, 2009.

[49] NL Ricker. Model predictive control of a continuous, nonlinear, two-phase reactor. *JOURNAL OF PROCESS CONTROL*, 3:109–109, 1993.

[50] Dorothy Denning. An intrusion-detection model. *Software Engineering, IEEE Transactions on*, SE-13(2):222–232, Feb. 1987.

[51] S. Joe Quin and Thomas A. Badgwell. A survey of industrial model predictive control technology. *Control Engineering Practice*, 11(7):733–764, July 2003.

[52] J.B. Rawlings. Tutorial overview of model predictive control. *Control Systems Magazine, IEEE*, 20(3):38–52, Jun 2000.

[53] T. Kailath and H. V. Poor. Detection of stochastic processes. *IEEE Transactions on Information Theory*, 44(6):2230–2258, October 1998.

[54] A. Wald. *Sequential Analysis*. J. Wiley & Sons, New York, 1947.

[55] Jaeyeon Jung, Vern Paxson, Arthur Berger, and Hari Balakrishan. Fast portscan detection using sequential hypothesis testing. In *Proceedings of the 2004 IEEE Symposium on Security and Privacy*, pages 211–225, May 2004.

[56] Stuart Schechter and Jaeyeon Jung Arthur Berger. Fast detection of scanning worm infections. In *Proc. of the Seventh International Symposium on Recent Advances in Intrusion Detection (RAID)*, September 2004.

[57] M. Xie, H. Yin, and H. Wang. An effective defense against email spam laundering. In *Proceedings of the 13th ACM Conference on Computer and Communications Security*, pages 179–190, October 30–November 3 2006.

[58] Guofei Gu, Junjie Zhang, and Wenke Lee. 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, February 2008.

[59] B.E. Brodsky and B.S. Darkhovsky. *Non-Parametric Methods in Change-Point Problems*. Kluwer Academic Publishers, 1993.